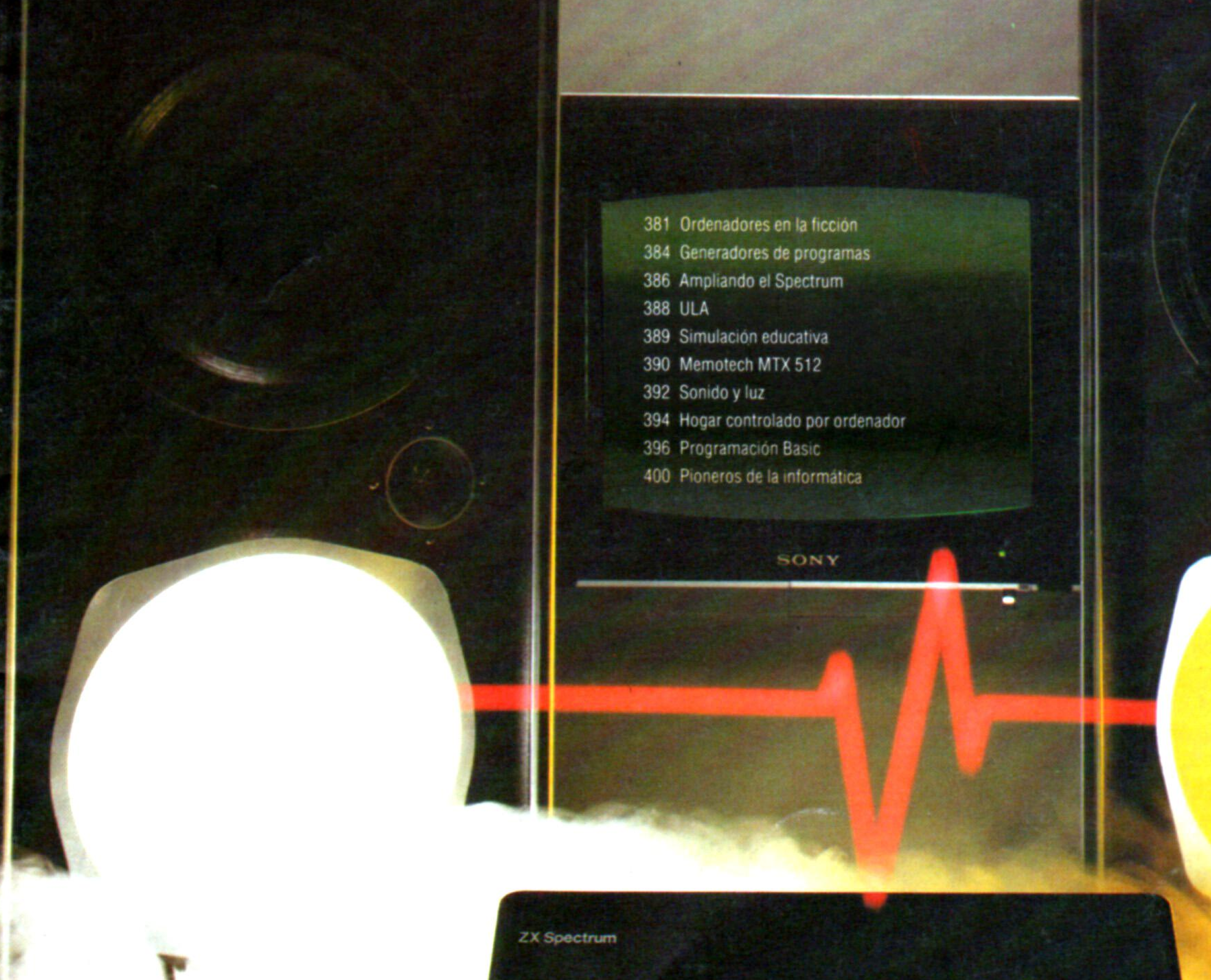
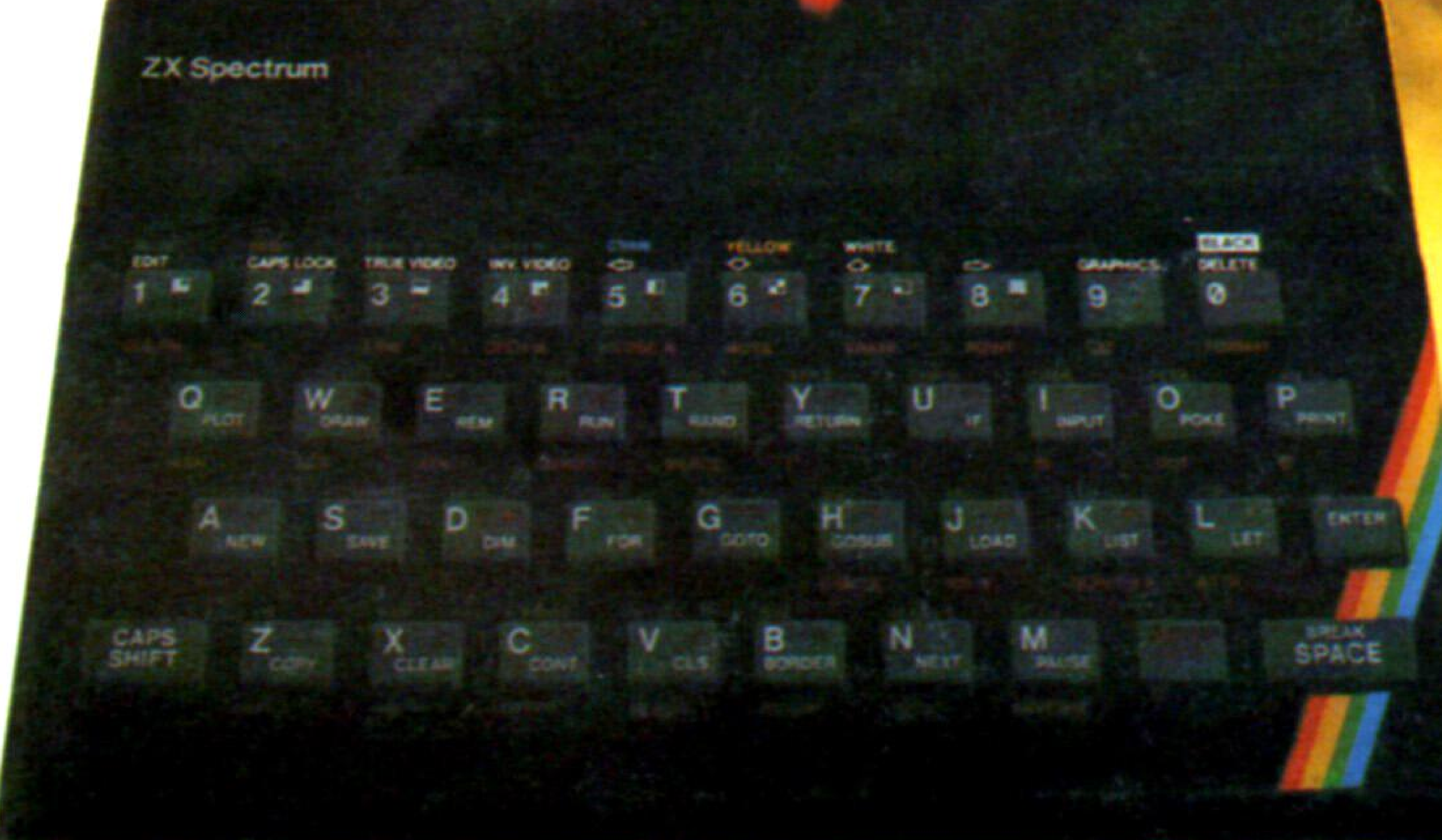
MI COMPUTER 20

CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR





00020
9 788485 822836

Editorial Delta, S.A.

150 ptas.

MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona, y comercializado en exclusiva por Distribuidora Olimpia, S.A., Barcelona

Volumen II - Fascículo 20

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones
Asesor técnico: Jesús Nebra

Redactores y colaboradores: G. Jefferson, R. Ford, S. Tarditti, A. Cuevas
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, Barcelona-8
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador,* se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-85822-90-0 (tomo 2)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 308405
Impreso en España - Printed in Spain - Abril 1984

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, Madrid-34.
Distribuye para Argentina: Viscontea Distribuidora, S.C.A., La Rioja 1134/56, Buenos Aires.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio Blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Ferrenquín a Cruz de Candelaria, 178, Caracas, y todas sus sucursales en el interior del país.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 16 690 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 087 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 3371872 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Distribuidora Olimpia (Paseo de Gracia, 88, 5.°, Barcelona-8), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

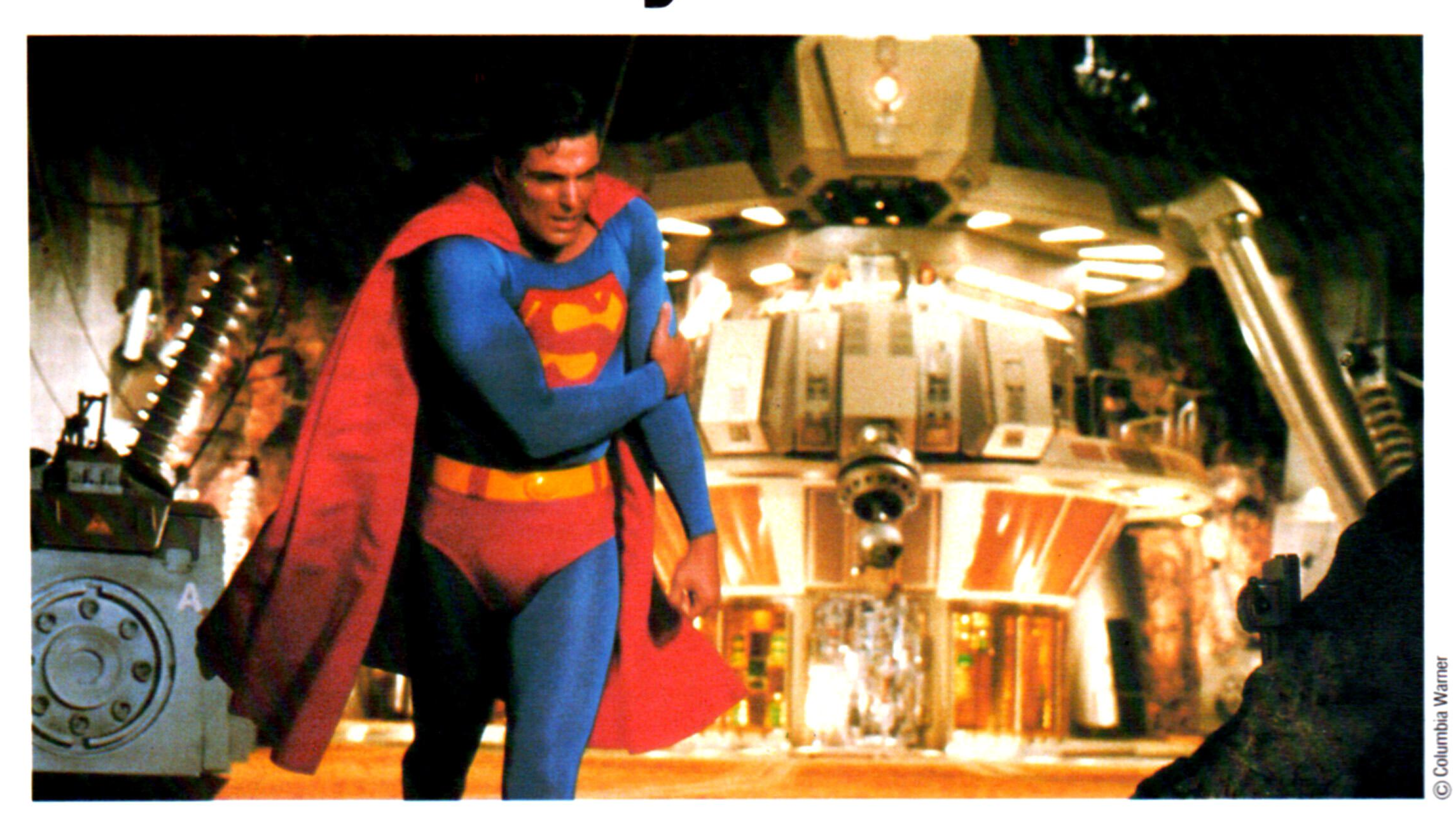
Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Distribuidora Olimpia, en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.



Informática y ficción



© Columbia Warner

Superman III
El fraude mediante ordenador es el tema central del tercer filme de Superman. Richard Pryor interpreta el papel de un malvado que se enriquece hurtando medio centavo en cada transacción a través del ordenador de un banco. Esta parte del guión se basa en varios casos reales de fraude. La película termina con la destrucción del mayor ordenador del mundo, que había sido construido únicamente con fines criminales

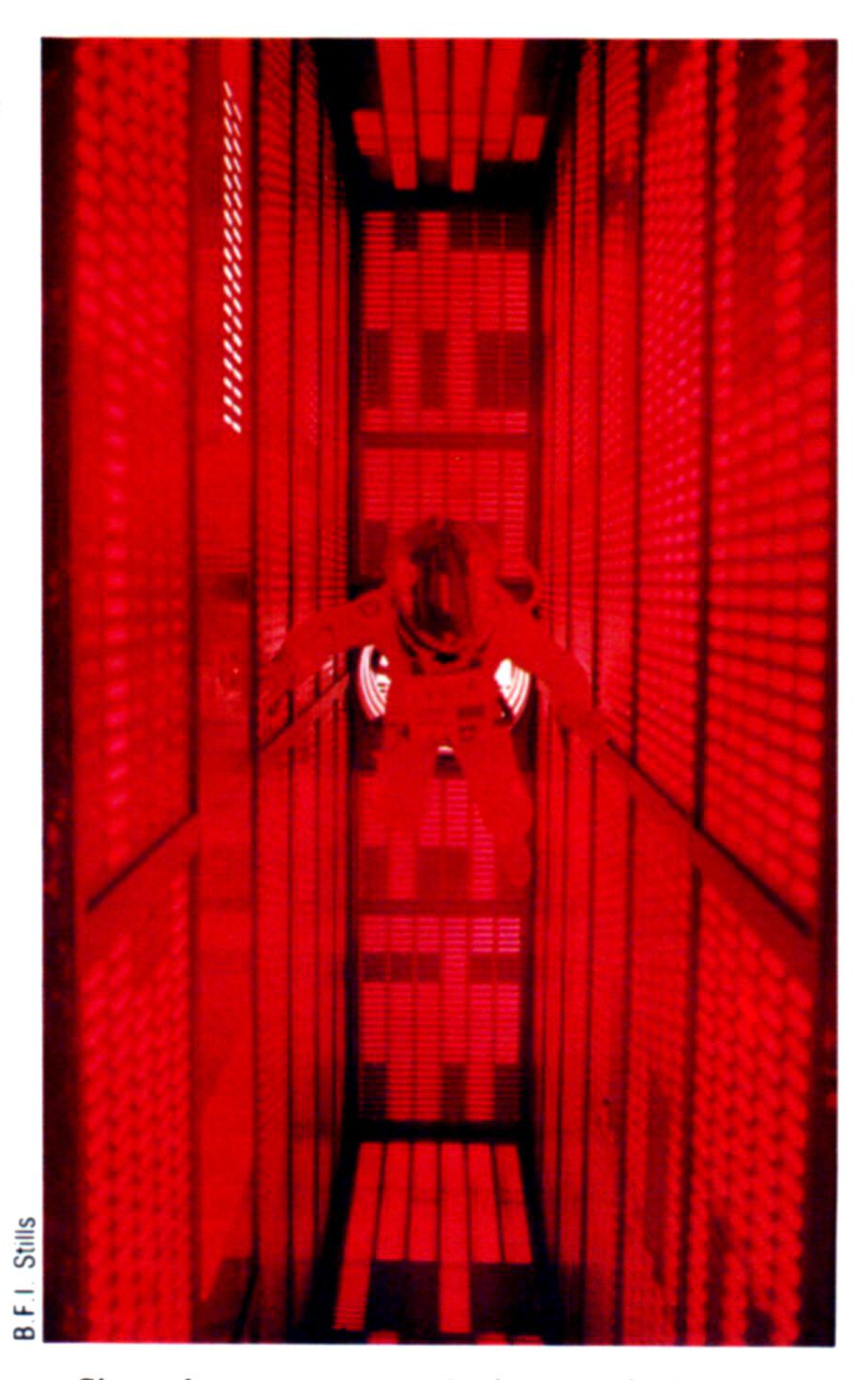
Los ordenadores se han usado a menudo en temas de ciencia-ficción, y algún escritor anticipó tecnologías hoy usuales

Muchos adelantos científicos y técnicos han sido imaginados por escritores de ficción y realizadores cinematográficos mucho antes de que fueran realmente factibles. Arthur C. Clarke, autor de *2001: una odisea del espacio (2001: a space odyssey)*, fue el primero en exponer la idea de la utilización de satélites geoestacionarios, en un artículo publicado en la revista *Wireless World*, a principios de la década de los cincuenta, es decir, con una antelación de casi veinte años sobre su puesta en funcionamiento. De una forma similar, en el relato de Robert Heinlein *Waldo*, se describía la manipulación de objetos por control remoto bastante antes de que se empezaran a emplear manos artificiales de robot. De hecho, numerosos inventores y científicos se han inspirado en las ideas surgidas de las mentes creadoras de escritores de obras de ficción y realizadores cinematográficos.

Sin embargo, frecuentemente, los ordenadores descritos en las obras de ciencia-ficción tienen poco que ver con la realidad. En la película futurista *Rollerball*, por ejemplo, se podía ver un ordenador que constaba de entrada con reconocimiento de voz y salida hablada, y cuya forma era semejante a un depósito para líquidos de estructura cúbica. En la realidad, por supuesto, el papel que desempeñan estas máquinas no suele ser muy espectacular ni interesante, aunque también en algunos filmes su única misión sea puramente decorativa, constituyendo un elemento más del mobiliario. No cabe la menor duda de que en los años sesenta y setenta, muchas películas en las que aparecían ordenadores con un diseño similar al que tienen los actuales han ayudado a educar al público, al mostrarle cómo eran en realidad estos nuevos, misteriosos, casi míticos «ordenadores».

La idea del ordenador empezó a tomar cuerpo en las novelas de ciencia-ficción poco después de que Charles Babbage (véase p. 220) comenzara su trabajo precursor sobre un ingenio analítico, a mediados del siglo XIX. En 1879, Edward Page Mitchell escribió un relato llamado *The ablest man in the world*, en el que se describe la implantación de una máquina calculadora en el cerebro de un deficiente mental, que lo convierte en un genio. Las ideas de Mitchell iban muy por delante de los avances científicos reales. En primer lugar, se centró en la idea de la miniaturización: la máquina computerizada es al mismo tiempo lo suficientemente pequeña como para ser colocada en el interior del cráneo del protagonista y lo suficientemente potente como para dotar a éste de una inteligencia superior. En segundo lugar, Mitchell prefiguró la idea de interconectar un ordenador con el cuerpo humano. Hoy en día, más de un siglo después de que fuera escrito este relato, se empiezan a perfeccionar las técnicas para conectar dispositivos electromecánicos de fácil control en el sistema nervioso central.

Por lo general, sólo unos cuantos escritores poseen conocimientos amplios de la arquitectura de los ordenadores, aunque algunos de ellos sean expertos ingenieros y muchos los utilicen (como procesadores de textos) en su trabajo. Del mismo modo, un escritor puede describir de manera convincente un viaje intergaláctico, sin necesidad de ser un astrofísico altamente cualificado o un experto en la tecnología de los cohetes. Asimismo, no existe ninguna razón por la cual no sea posible especular sobre las posibles características de las futuras generaciones de ordenadores aunque no se posea un conocimiento profundo de las máquinas que se emplean en la actualidad.



B.F.I. Stills

Ordenador HAL
El ordenador Heurístico y ALgorítmico de la obra de Arthur Clarke *2001: una odisea del espacio* es un buen ejemplo de la omnipotente máquina ordenadora que aparece con frecuencia en los filmes de ciencia-ficción. En este caso, HAL es el responsable de los detalles de los objetivos de la misión mientras que los miembros humanos de la tripulación desarrollan una misión secundaria, lo cual conduce a la máquina a creer que puede prescindir de éstos. Se supone que Clark eligió las iniciales HAL por su proximidad en el orden alfabético con IBM

Sin embargo, estas teorías han conducido al concepto de un ordenador de ciencia-ficción estándar que posee, en comparación con las posibilidades reales de los ordenadores actuales, una gama de propiedades que son imposibles de lograr. Para empezar, este ordenador-tipo almacena en su memoria todos los datos e información existentes en el mundo y cualquier idea que haya imaginado la mente humana, y es capaz de recuperar el dato necesario instantáneamente, mediante un procedimiento similar al funcionamiento del cerebro. El ordenador HAL de *2001: una odisea del espacio* es una de estas máquinas superinteligentes.

Otra de las características con que dotan los escritores de ciencia-ficción a sus ordenadores es la omnipresencia, si bien, al mismo tiempo, se crea la sensación de que muy pocas personas tienen acceso a ellos. Dos de los requisitos básicos del superordenador son la capacidad de hablar (pero sin hacer ninguna mención a que se puede tratar simplemente de un producto sintético de ensamblamiento de fonemas), y el reconocimiento de la voz (que indefectiblemente tiene en cuenta las peculiaridades del habla de cada persona). Pero también el reconocimiento visual de los objetos y la capacidad de sintetizar alimentos (quizá a partir de sus componentes elementales básicos) constituyen a su vez otras propiedades de gran utilidad.

El superordenador que hemos descrito en términos generales, en la mayoría de los casos está dotado asimismo de atributos humanos, y esta caracterización le hace aparecer como una especie de ser superior. Sin embargo, algunas veces el papel que desempeña puede ser decididamente malévolo. En el filme *Dark star*, por ejemplo, una bomba controlada por ordenador adquiere las características perturbadas de un asesino psicópata. Cuando es representado de esta forma, el superordenador, ciertamente, forma parte del reino de la fantasía, pero, al mismo tiempo, en los componentes de la informática actual es posible entrever las características básicas de esas propiedades que hoy son sólo atributo de la ficción.

En estos momentos ya existen máquinas con una gran capacidad de memoria y que requieren un tiempo de acceso muy pequeño. A principios de los ochenta se crearon memorias que alcanzaban el valor del gigabyte (mil millones de bytes), y las máquinas comerciales más rápidas poseían una velocidad de procesamiento superior a los diez millones de instrucciones por segundo. Volviendo otra vez al campo de los ordenadores que hablan, se está muy cerca de alcanzar el nivel de perfección mostrado en las películas. La calidad que pueda poseer esta característica depende sencillamente del espacio de memoria disponible, de la velocidad de procesamiento y del tiempo de programación. En cambio, el reconocimiento de voz es más difícil de lograr, debido a la amplia gama de formas del habla humana. Dos personas pueden hablar el mismo idioma, pero para el ordenador es como si se expresaran en lenguas completamente diferentes.

El reconocimiento visual de objetos se encuentra también en una fase muy elemental de desarrollo, pero la tecnología ligada a esta esfera está avanzando rápidamente. Cuando analizamos los robots industriales (véase p. 281), nos dimos cuenta de que se ha producido un gran adelanto en el reconocimiento de objetos mediante cámaras de televisión con un dispositivo de carga en paralelo, y vimos que el robot era capaz de elegir un objeto determinado de entre varios dispuestos aleatoriamente. Un reconocimiento visual que sea significativo depende del tamaño del vocabulario óptico, el cual, a su vez, está en función de la capacidad de memoria y de la potencia de procesamiento. En lo referente a la síntesis de alimentos, quizá no sea posible hacer que un plato tenga la apariencia de filete con patatas o pescado frito, pero, sin lugar a dudas, existe la posibilidad de hacer que tenga el aroma y el sabor de éstos, aun cuando los ordenadores no puedan, todavía, crearlos a partir de sus elementos.

No todos los autores llegan al extremo de atribuir poderes extraordinarios a sus máquinas de ficción. John Brunner, por ejemplo, en su novela de ciencia-ficción *Situación en Zanzíbar (Stand on Zanzibar)*, publicada en 1969, describe el mundo tal como podría ser en el año 2010, con unos problemas de superpoblación y de falta de alimentos que habrían alcanzado niveles insostenibles. El ordenador que describe, al que llama «Shalmaneser»,

tiene, por descontado, una capacidad de memoria y una velocidad de procesamiento considerables (puesto que está conectado a todos los aparatos de televisión de la Tierra), pero su lenguaje es muy similar a los que se utilizan en la actualidad:

PROGRAMA RECHAZADO
Q razón del rechazo
ANOMALIAS EN DATOS BASICOS
Q definir Q especificar
INFORMACION NO ACEPTABLE EN LAS
SIGUIENTES CATEGORIAS HISTORIA COMERCIO
INTERACCION SOCIAL CULTURA
Q aceptar datos como están
PREGUNTA SIN SENTIDO E IMPRACTICABLE

Evidentemente, Brunner llegó hasta tal punto en la utilización de un lenguaje cercano al normal que un analista lo podría tomar como una respuesta de un sistema realmente existente. Sus otras predicciones son asimismo convincentes, y no es sorprendente que la novela ganara los premios más importantes del género de la ciencia-ficción.

En filmes y novelas más recientes, los ordenadores se han convertido en algo más que una parte del mobiliario o incluso que personajes secundarios, y en la actualidad frecuentemente constituyen la trama central del argumento. Un buen ejemplo de ello lo constituye *TRON*, producción de la firma Walt Disney. Anteriormente ya hemos hablado de este extraordinario largometraje (su nombre deriva de un sistema de realización *TRace ON*), presentándolo como representativo del sistema de realización de dibujos animados mediante ayuda de ordenadores (véase p. 181). La acción se desarrolla en el mundo real y en el *interior* de un ordenador. El mundo exterior está formado por personajes tales como ingenieros en software, programadores de sistemas y otras personas relacionadas con el mundo de la informática. Pero en el interior de la máquina, las secciones independientes del programa y del sistema son los verdaderos protagonistas, y la arquitectura de la máquina es el escenario en el cual tiene lugar la acción.

También existen obras de ficción en las que, en realidad, no se hace mención de los ordenadores, pero en las cuales el lector no tiene la menor duda de que sin esta maquinaria informática extremadamente poderosa, la situación que se describe nunca podría existir. Ejemplos notables entre estas últimas son *1984*, de George Orwell, y *Un mundo feliz (Brave new world)*, de Aldoux Huxley. Ambas novelas se desarrollan en una época futura con respecto al tiempo en que fueron escritas, y en un mundo totalmente dominado por una pequeña camarilla que reprime al resto de la población. Quizá deberíamos fijarnos en estos dos libros para formarnos una idea de las consecuencias que podría acarrear una mala utilización del poder de la informática.

Es totalmente imposible agotar el tema en este análisis del ordenador como argumento de ciencia-ficción, pero es necesario mencionar algunas novelas que describen ideas especialmente ingeniosas. Por ejemplo, la obra de John Barth *Giles Goat-Boy*. Esta voluminosa novela (812 páginas en su edición inglesa en rústica) se centra en la premisa de que el libro es obra de un superordenador llamado WESCAC, y relata un incidente que le sucedió a su "autor".

Por último, deberíamos recordar también que no es únicamente en la esfera de la ficción donde se encuentran creaciones acerca de los ordenadores. De los miles de libros que se han escrito centrados en el tema específico de los ordenadores y de la informática, hay uno que sobresale del resto simplemente por su calidad narrativa. La obra de Tracy Kidder *Soul of a new machine* es la historia del desarrollo del Data General's Eagle, un miniordenador de 32 bits. Si bien el tema trata, aparentemente, de los ingenieros y directores que participaron en el proyecto, no cabe duda de que el protagonista es el propio ordenador.



Juegos de guerra
Inconscientemente, un adolescente que intenta poner en comunicación su ordenador con el de un amigo, a través de la red telefónica, interfiere el ordenador principal de defensa de la OTAN. Creyendo que lo que ve en su pantalla es un juego, empieza a jugar, hasta descubrir que ha desencadenado la tercera guerra mundial...

Autor original

Es posible escribir programas que generen por sí mismos otros programas o que corrijan errores humanos en la codificación

"Si los ordenadores son tan inteligentes, ¿por qué necesitan ser programados por los seres humanos?" Las personas con experiencia en informática tenderán a restar importancia a una pregunta de este tipo, hecha por un principiante escéptico, pero no es tan banal como pudiera parecer a simple vista. En la actualidad se realizan numerosas investigaciones con la finalidad de escribir programas que puedan generar otros programas, y sistemas operativos capaces de corregir errores en el código escrito por el operador.

Téngase en cuenta, por ejemplo, el mensaje ¿ERROR SINTAXIS?, con que se encuentran frecuentemente los que utilizan los ordenadores personales. Esto puede resultar bastante desconcertante porque el mensaje suministra muy poca información. Un compilador de un ordenador de mayor capacidad daría normalmente muchos más datos, como podría ser la naturaleza del error que se ha hallado. Por ejemplo, el mensaje indicando el error podría tener la siguiente configuración:

```
1090 LET A = (C*2+F$)*((FG-C)*TH+1))
                  ↑                ↑
ERRORES:1) MAL EMPAREJADO — VARIABLE
             ALFANUMERICA F$ NO PERMITIDA
        2) ULTIMO CIERRE PARENTESIS NO
             ESPERADO
```

No existe ninguna razón fundamental por la cual estas técnicas no puedan utilizarse en un intérprete de un ordenador personal, ya que el coste del ROM suplementario que sería necesario para poder almacenar las rutinas correspondientes sería mínimo. No obstante, muy pocos lo emplean, así como tampoco utilizan procedimientos elementales de control de error: la mayoría de ellos no verifican la sintaxis del código cuando éste es introducido. Sin embargo, con frecuencia existe la posibilidad de comprar chips de ROM suplementarios o cartuchos de software que amplíen la gama de órdenes disponibles en BASIC, en particular aquellas que tienen relación con el desarrollo y la eliminación de errores en los programas. Estas órdenes BASIC comprenden:

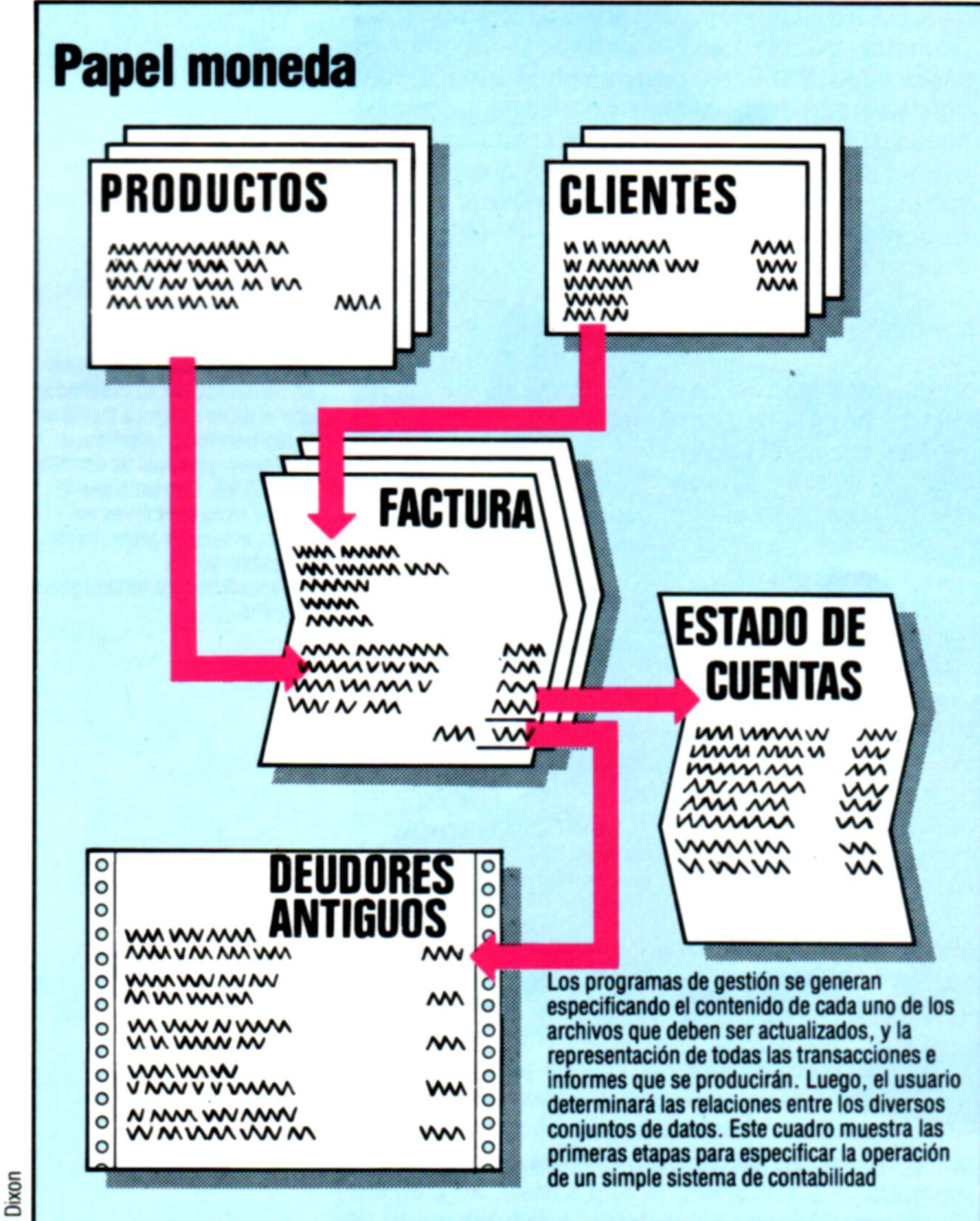
HELP: imprime la línea del programa y subraya la posición exacta del carácter en donde termina la realización del programa. Esto indicará normalmente, aunque no siempre, el origen del error de sintaxis.

DUMP: imprime una lista de todos los nombres de variables y de sus contenidos utilizados en ese momento por el programa. Esto es útil para deducir hasta qué punto se ha producido el procesamiento del programa antes de ocurrir el error.

TRACE: visualiza (en una ventana en la esquina de la pantalla) el número (o números) de línea que se está procesando mientras el programa está en funcionamiento. Esto ayuda al usuario a trazar el organigrama del programa, y asegura, entre otras cosas, que las subrutinas están siendo ejecutadas en el orden deseado.

La escritura de programas que permitan a un ordenador corregir errores humanos de codificación no es, en general, una tarea sencilla. Sin embargo, en el caso de algunos errores específicos sí resulta bastante simple. Por ejemplo, sabemos que todas las líneas del programa tienen que empezar con una contraseña en BASIC (si bien algunas máquinas permiten omitir la palabra LET). Por tanto, si una línea empieza con PRUNT o PRONT, será fácil deducir que debería decir PRINT.

Pero para otro tipo de errores más complejos que estos procedimientos básicos descritos, la corrección automática presenta un grado de dificultad muy superior. En el ejemplo que hemos dado ante-



Los programas de gestión se generan especificando el contenido de cada uno de los archivos que deben ser actualizados, y la representación de todas las transacciones e informes que se producirán. Luego, el usuario determinará las relaciones entre los diversos conjuntos de datos. Este cuadro muestra las primeras etapas para especificar la operación de un simple sistema de contabilidad

riormente, ¿es F$ un error de impresión y debería decir F o FS o F4? ¿O algo totalmente distinto? Si tuviera que enseñar el listado a otro programador experimentado, éste debería ser capaz de identificar las faltas y efectuar las correcciones. Para tomar sus decisiones, tendría que basarse en dos criterios: el contexto en el que ha aparecido la línea del programa, y su propia experiencia.

Por extraño que pueda parecer, esta técnica ha sido aplicada con mayor frecuencia para corregir el texto que para la verificación del código del programa. Un paquete para comprobar la escritura correcta de las palabras, por ejemplo, repasará todo el texto e indicará los términos que no se corresponden con las entradas de su diccionario, el cual puede constar de unas 50 000 voces, registradas en un disco. La mayoría de estos paquetes es capaz de aprender nuevas palabras (tales como nombres propios o de compañías) y de añadirlas a sus diccionarios. Algunos de ellos, de características más complejas, pueden incluso sugerir la forma correcta de escribir una palabra en el caso de que se detecte un pequeño error de escritura. También se han desarrollado procesadores de textos que tienen la posibilidad de aplicar estos mismos procesos a la corrección ortográfica y de estilo, suministrando indicaciones tales como puntuación incorrecta, repetición de una misma palabra dentro de un párrafo, metáforas confusas y adjetivos inadecuados.

Sin embargo, la mayor parte de las investigaciones se han centrado en el desarrollo de sistemas capaces de *crear* programas, en vez de *corregir* los existentes. En 1981 se anunció la salida al mercado de un producto de software que hizo estallar una de las batallas más enconadas jamás libradas en la industria del microordenador. Con un gran sentido de la oportunidad, recibió el nombre de "El último", y fue presentado con la pretensión de ser un programa que podía escribir cualquier otro que se deseara; por lo tanto, se constituía en el último programa que se necesitaría comprar. Esto resultó ser un reclamo publicitario que no se ajustaba a la realidad, aunque "El último" era una ayuda muy valiosa para el desarrollo de cierto tipo de programas, principalmente aplicaciones comerciales. En la actualidad existen en el mercado algunos de estos productos, de los cuales sólo una mínima parte está destinada a la gama de los ordenadores personales. A estos productos se les designa colectivamente con el nombre de *generadores de programas*.

Veamos ahora en qué consiste el concepto básico de un programa que sea capaz de escribir otros. Consideremos este ejemplo trivial:

```
10 PRINT "¿QUE DEBE VISUALIZAR EL PROGRAMA
   EN LA PANTALLA?"
20 INPUT A$
30 PRINT "EL PROGRAMA ES:"
40 PRINT "10 PRINT";CHR$(34);A$;CHR$(34)
```

Si a esta pregunta se responde con HOLA, el programa (que debe funcionar en la mayoría de los ordenadores personales) debería imprimir la línea:

```
EL PROGRAMA ES
10 PRINT "HOLA"
```

Si se aplica la misma técnica a las fases de entrada, cálculo y salida de la aplicación que se tiene en mente, se podría escribir un generador de programas muy simple. Si todas las preguntas que efectúa el ordenador son expresadas claramente, una persona que no tuviera experiencia previa podría desarrollar, con el generador, un programa sencillo.



Herramientas del oficio

"El juego de herramientas del programador" para numerosos ordenadores personales, puede encontrarse en el mercado en forma de chips de ROM o cartuchos. Permite ampliar la gama de órdenes en lenguaje BASIC, especialmente para escritura de programas y eliminación de errores

Kevin Jones

Los generadores de programas producidos comercialmente emplean una técnica semejante. La mayoría de las aplicaciones de gestión consiste en una combinación de cinco procesos distintos: entrada de datos, salida a pantalla o impresora, almacenamiento en un archivo de datos, recuperación de información y cálculo. El generador tendrá unas subrutinas normales y muy flexibles para cada uno de estos procesos. Al pedir que se especifique la estructura exacta de los datos, y cómo deben aparecer en la pantalla y en la impresora, se hará que el generador cambie los valores de algunas variables de las subrutinas y las relacione entre sí para crear el programa.

Aunque los generadores de programas cada vez se vuelvan más sofisticados, no serán capaces de sustituir, en un futuro inmediato, a los programadores humanos, debido a que se encuentran limitados por los siguientes factores. Primero, la técnica descrita es muy adecuada para aplicaciones comerciales tales como contabilidad o control de stocks, pero generalmente estos generadores de programas no pueden aplicarse en el procesamiento de textos o en los programas para juegos. En segundo lugar, debido a que el generador de programas debe hacer uso de las subrutinas flexibles normales, el listado resultante no será tan eficiente (desde el punto de vista de la velocidad y de la memoria utilizada) como si hubiera sido escrito con esa finalidad por un programador. En tercer lugar, los programas producidos por los generadores no suelen resultar tan cómodos de utilizar como los sistemas que realizan hoy los programadores humanos.

Por último, los generadores de programas existentes hoy en día, en la práctica sólo son capaces de sustituir la última etapa de la programación: la escritura del código. El usuario todavía tiene que esforzarse en pensar la disposición exacta de los datos, entrada y salida que serán necesarios para que funcione el programa. Generalmente, las primeras etapas de la programación son las más difíciles de desarrollar, y requieren que, además de tener un conocimiento profundo de programación, se posean también otras cualidades. Las compañías más importantes disponen de especialistas, llamados *analistas de sistema*, para especificar los programas que necesitan, y estas especificaciones son codificadas por los programadores. Para poder crear un programa de ordenador, los generadores de programas aún deben adquirir estas cualidades.

Sobre el arco iris

Algunos ordenadores pueden adquirir mayor complejidad gracias a una serie de complementos. Ampliemos al máximo un Spectrum

Acoplador acústico
El Micro-Myte 60, representado aquí, permite poner en comunicación dos ordenadores

Microdisco
El microdisco usa cartuchos que contienen un bucle cerrado de cinta magnética; cualquier punto dado de éste pasa por el cabezal de lectura-escritura cada siete segundos. La transferencia de información se produce a 6 Kbytes por segundo (cuatro veces la velocidad de un reproductor normal de cassette). Tal como vemos, se pueden conectar conjuntamente hasta ocho microdiscos, que proporcionan una capacidad total de 700 Kbytes o más

Teclado
El teclado FDS de Fuller posee teclas para funciones y un espaciador que abarca toda su anchura

Palancas de mando
Mediante la interface 2 de Sinclair puede acoplarse cualquier palanca de mando que emplee la interface Atari, sin que haya que tener en cuenta su sistema de funcionamiento. La interface 2 puede conectar dos palancas a un tiempo

Paquete RAM
El Spectrum de 16 Kbytes puede ampliarse mediante la adición de un cartucho suplementario de 32 Kbytes

Cuando fue presentado a principios de 1982, el Sinclair Spectrum fue recibido como un adelanto decisivo, tanto desde el punto de vista de su rendimiento como de su precio. En 1983, su primer año completo de producción, las ventas del Spectrum (600 000 unidades) representaban más de la mitad del número total de ordenadores personales comercializados en Gran Bretaña, éxito sorprendente incluso para los mismos fabricantes. El Spectrum constituía, ciertamente, una mejora considerable con respecto al anterior modelo de Sinclair, el ZX81, con 16 o 48 Kbytes de RAM como capacidad normal de memoria; ocho colores para márgenes, fondo y texto, y una capacidad limitada de resolución gráfica; un teclado mejorado, pero todavía poco eficiente; y la posibilidad de generar sonidos simples. Pero todas estas características diferentes no fueron impedimento para que los fabricantes independientes produjeran una amplia gama de accesorios. La misma Sinclair no ha permanecido inactiva en este aspecto, añadiendo almacenamiento secuencial, en forma de microdisco, e interfaces para la conexión de cartuchos de ROM y palancas de mando para juegos.



Impresora en serie
Además de la impresora ZX, cabe la posibilidad de montar una matriz de puntos convencional o una impresora margarita, utilizando una interface RS232 o Centronics. La interface 1 de Sinclair admite dispositivos en serie

Monitor TV
Como no se puede adaptar un monitor a la salida del Spectrum, es difícil obtener gráficos de alta calidad. En la figura se muestra el Profeel de Sony, sistema de televisión modular que tiene el receptor separado del monitor

Sintetizador de música
El Trichord de Petron constituye un buen ejemplo de un tipo de sintetizador de música adecuado para el Spectrum. El Trichord es una unidad de tres voces que ofrece hasta 6134 acordes de tres notas si se utiliza con un Spectrum de 48 Kbytes

Reproductor cassette
Incluso con el microdisco montado, el Spectrum puede escribir o leer conectado a una grabadora de cassette doméstica. Por lo tanto, se puede utilizar el software comercial en cassette

Sintetizador de voz
El Cheetah Sweet Talker es un sintetizador que emplea el sistema alofónico simplificado para formar palabras a partir de las sílabas que las componen. Sesenta y tres alófonos y cuatro intervalos de espacio de longitudes variables constituyen esta unidad

Interface 2
Algunas compañías independientes han producido interfaces para palancas de mando programables, aunque el mismo modelo Sinclair incorpora una conexión ROM, de forma que los juegos y lenguajes alternativos sean más fáciles de montar y usar

Teclado
Si bien el teclado estándar del Spectrum está muy mejorado con respecto a los que incorporaban los ZX80 y ZX81, éste continúa teniendo una configuración inadecuada

Interface 1
Diseñada específicamente para servir al micromando, la interface 1 también contiene un sistema de circuitos que permite usar varios dispositivos, como impresoras, plotters o modems. Además, hace posible conectar diversos Spectrum entre sí en una red local

Impresora ZX
Con el fin de producir una impresora de bajo coste, suficiente para cubrir las necesidades básicas de un usuario de ordenador personal (primordialmente listados de programas), Sinclair ha creado una unidad que emplea una forma de erosión por chispa para imprimir sobre un papel de base alumínica

Diseñada a medida

La ULA (disposición lógica no comprometida) controla todas las funciones de un ordenador personal, salvo la CPU, ROM y RAM

De los numerosos avances producidos en el diseño electrónico, a consecuencia del gran desarrollo que ha tenido lugar en la industria del microordenador, uno de los más significativos ha sido la realización de un tipo de chip que recibe el nombre de *disposición lógica no comprometida* (*Uncommitted Logic Array*: ULA), con el cual es posible construir ordenadores muy complejos, así como otros dispositivos, con sólo cuatro componentes: CPU, RAM, ROM y —para unir a los tres— una ULA.

¿Y qué es una ULA? Tal como sugiere su nombre, consta de un gran número (una disposición) de puertas lógicas, las cuales, inicialmente, no están comprometidas pero pueden ser modificadas para realizar casi todas las operaciones que necesita el diseñador. La ULA puede considerarse como un desarrollo de la ROM, puesto que el contenido de ambas sólo puede ser determinado por el fabricante del chip, y no por el usuario.

Antes de que una ROM o una ULA sea "programada", ésta únicamente consiste en un gran número de circuitos (o celdas) electrónicos sencillos, que no están conectados y, por tanto, no pueden realizar ninguna acción. Todos los chips se construyen por la superposición de capas de materiales semiconductores (véase p. 122). La última capa, normalmente, está integrada por un material conductor, y forma las conexiones entre las diversas celdas. Lo que proporciona a la ULA su gran flexibilidad es la gran variedad de interconexiones posibles; y si bien la constitución de cada celda es bastante simple, consistiendo nada más que en un par de transistores o en una sola resistencia, pueden conectarse entre sí mediante la capa final y formar circuitos bastante complicados: un flip-flop, por ejemplo (véase p. 305).

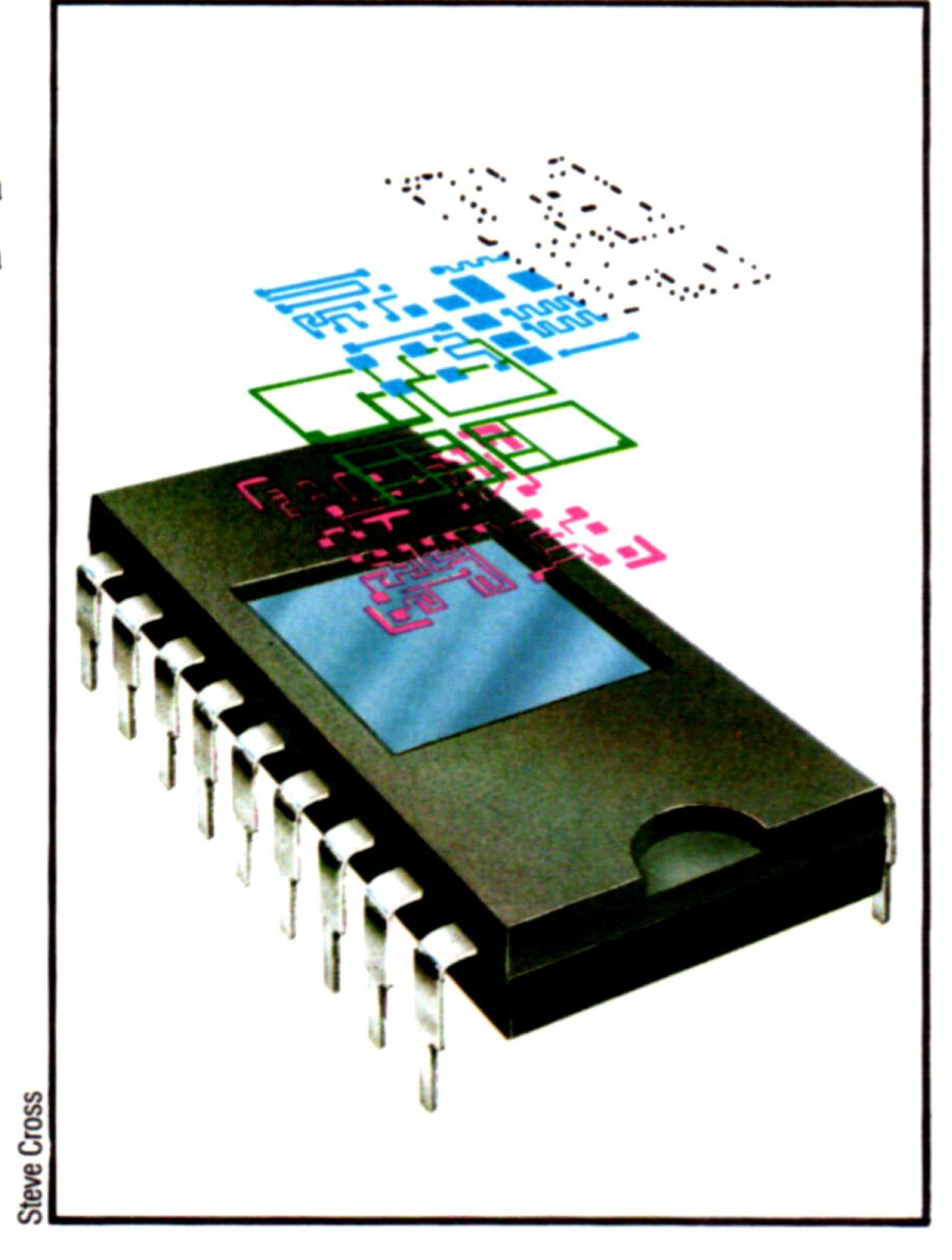
Estos circuitos, llamados *módulos*, se pueden construir con menos de media docena de celdas, y puesto que una ULA de gran tamaño puede tener varios miles de celdas, los mismos módulos pueden conectarse entre sí para constituir circuitos complejos, tales como registradores, contadores o circuitos de medición.

Una ULA puede programarse para que realice una gama extremadamente amplia de actividades. Puede utilizarse como sintetizador de sonido, o para controlar la exposición, el enfoque y el motor de una cámara, o para efectuar la mayoría de las funciones de un termómetro digital. Y, aparte de la ULA, apenas es necesario otro circuito externo.

Tal como se podía esperar, en el proceso de diseño de la capa que conecta las células de una ULA se emplean ampliamente los ordenadores. Un miniordenador como el DEC PCP11/23 puede hacer funcionar un sistema de diseño auxiliado por ordenador y crea en primer lugar un diagrama codificado de la lógica deseada. Luego, el sistema traza, y asimismo codifica, un mapa del esquema planeado. Esto es efectuado por un terminal de gráficos, y en un plotter se puede producir una copia.

Una vez se ha completado el diseño, éste es transmitido a un ordenador de mayor capacidad, el cual verifica que el plan es aceptable, lo compara con el diseño lógico original y comprueba que no tenga ningún error de importancia. Luego, es sometido a otro programa que simula el circuito que resultaría, utilizando un programa piloto suministrado por el cliente. Una vez finalizado el diseño, el ordenador puede producir la plantilla óptica que se emplea para construir la capa final.

¿Hasta dónde pueden llegar las ULA? La idea de colocar un gran número de circuitos sencillos de silicio y permitir que el usuario decida cómo deben actuar es tan atractiva que se puede convertir en el método aceptado de realizar la mayor parte de los sistemas de circuitos. Sin embargo, en el nivel presente de la tecnología, las ULA únicamente son rentables cuando se necesitan, como mínimo, unos dos mil circuitos idénticos. Las PROM (*Programmable Read Only Memory*: ROM programable), EPROM (*Erasable PROM*: PROM que puede borrarse); EEPROM (*Electrically Erasable PROM*: PROM que puede borrarse eléctricamente) y EAROM (*Electrically Alterable ROM*: ROM alterable eléctricamente) son todas alternativas de la ROM, que pueden ser programadas por el usuario mediante un equipo adecuado.


Steve Cross

ULA
Todos los chips semiconductores se construyen a partir de capas de depósitos de semiconductores, que son tratados químicamente de forma individual para crear los elementos del circuito. La última capa determina la conexión entre los elementos. Una ULA consiste en una disposición de elementos lógicos que pueden combinarse para formar un circuito lógico complejo

Frente a la realidad

La simulación es una de las mejores aplicaciones educativas de los ordenadores. Veamos con detenimiento algunos programas

Vientos

Winds (Vientos), programa de simulación realizado por la Longmans para el BBC Micro, parte del supuesto de que la persona que lo está haciendo funcionar es el patrón de una vieja goleta. En la pantalla del monitor aparece un mapa del globo terráqueo, en el cual el navío está representado por un pequeño punto y se gobierna mediante una brújula. Se eligen los puertos de salida y llegada y la fecha en que empieza la travesía. La velocidad del barco depende de la dirección y la intensidad de los vientos dominantes, que aparecen representados en la parte inferior de la pantalla. Otros parámetros representados en el monitor son: la posición de la goleta, indicada por la longitud y la latitud; la zona de vientos (alisios, p. ej.); la fecha; la distancia ya recorrida, y la duración total de la travesía.

Tomemos la ruta más directa desde Londres hasta Río de Janeiro, y supongamos que se zarpa el primero de enero. El barco se dirige hacia el sur y avanza rápidamente gracias a los vientos dominantes en la zona, hasta llegar a la línea del ecuador. Aquí, debido a la zona de las calmas ecuatoriales, no progresa durante tres días. Finalmente, empiezan a soplar los vientos alisios del sudoeste, pero esto crea un problema: ¿cómo navegar en dirección sudoeste con vientos procedentes de este mismo cuadrante? La solución es navegar en zigzag (o en "virada"), primero hacia el sur y luego hacia el oeste, hasta alcanzar Río de Janeiro, después de recorrer 15 480 km durante 207 días.

Otros viajes pueden estar erizados de peligros: huracanes, hielos polares, naufragios, pueden ser algunos de los contratiempos que haya que afrontar. Este programa de simulación se puede utilizar de varias formas distintas. La más sencilla de ellas podría consistir en emplearlo a modo de juego educativo para enseñar a niños los puntos cardinales de la brújula.

Simulación de vuelo

Flight simulation (Simulación de vuelo) es una versión para ordenadores personales de un juego recreativo, en el cual uno mismo es el piloto de un pequeño aeroplano. La pantalla muestra la visión que tiene el piloto desde la cabina del avión, con el cuadro de instrumentos en la parte inferior de aquélla. Éste está compuesto por diversos aparatos de medición, diales y testigos ópticos, que deben ser observados atentamente con el fin de pilotar el avión de manera correcta. La columna de mando del aeroplano está representada por las cuatro teclas del ordenador marcadas con una flecha; los otros controles (potencia, equipo para ayuda en el aterrizaje, etc.) también son manejados a través del teclado.

Sin embargo, de forma diferente que en los juegos recreativos, aquí se tiene la oportunidad de familiarizarse con los mandos, empezando a pilotar con el avión ya en el aire. Para obtener un conocimiento de las características del vuelo, se puede visualizar en la pantalla un plano que indica la posición del aparato, de los aerofaros de navegación y de las pistas de aterrizaje. La mejor forma de navegar es no perder de vista alguno de los aerofaros de navegación, y luego ladear el avión hasta que quede alineado con éste. Esto es indicado por el punto luminoso intermitente del "Reloj RDF", que se desplaza hasta alcanzar la parte superior del dial. Entonces se debe volar en línea recta hasta estar a la altura del aerofaro. Mediante este método, resulta fácil llegar hasta el aeropuerto.

La maniobra más difícil de efectuar es el aterrizaje. Es necesario estar perfectamente en línea recta con la pista y aproximarse a la velocidad, altura y ángulo exactos. Aun en el caso de que todos estos factores sean correctos, se puede cometer un fatal error: ¡olvidarse de bajar el tren de aterrizaje!

Simulación fisiológica

En *Physiological simulation* (Simulación fisiológica) se representa el cerebro humano y la misión consiste en ¡mantener con vida al cuerpo durante 50 minutos! El programa simula los diversos cambios fisiológicos (temperatura del cuerpo, pérdida de agua, etc.) que se producen cuando el cuerpo humano realiza diversas actividades. Lo primero que hay que hacer es introducir en el ordenador la edad y el sexo de la persona y la actividad que se desea realizar. La más fácil de simular es dormir, puesto que el cuerpo gasta poca energía. Otras pueden consistir en andar, subir una montaña y, la más agotadora de todas, correr.

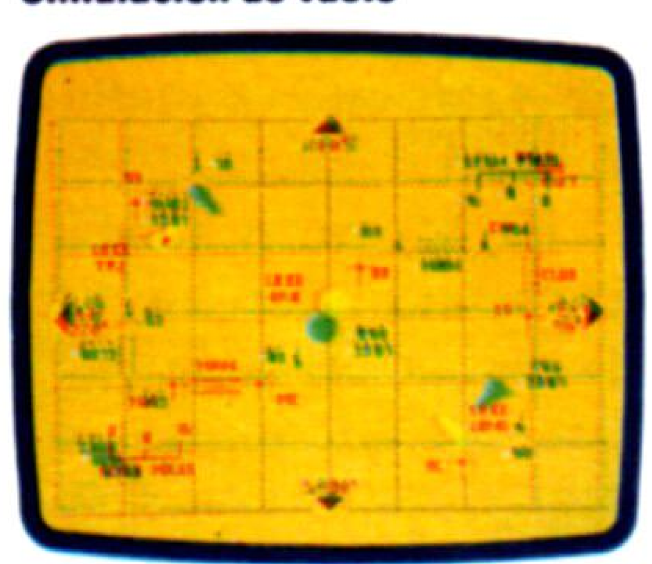
Los parámetros que se deben controlar son: la capacidad torácica, el ritmo respiratorio y la transpiración. Hay que elegir unos valores iniciales; por ejemplo, una capacidad torácica de 2,5 litros, un ritmo respiratorio de 15 veces por minuto, y una cantidad de sudor de tres gramos por minuto, y a partir de aquí se puede empezar la simulación.

En la pantalla se ven cinco gráficos de las diversas funciones corporales, y un reloj. A medida que va transcurriendo el tiempo, los gráficos suministran datos de cómo se está desarrollando la actividad elegida, y debe evitarse que cualquiera de ellos alcance un nivel peligroso. Si, por ejemplo, la temperatura del cuerpo aumenta excesivamente, se puede suspender la actividad durante un corto período de tiempo e incrementar el régimen de transpiración para intentar que vuelva a bajar. Si esta tentativa resulta infructuosa y un gráfico sobrepasa el nivel de peligro, se puede recibir un diagnóstico conciso como "LA PERSONA ESTA MUERTA".

Vientos



Simulación de vuelo



Simulación fisiológica



Ian McKinnell

Memotech MTX 512

Este ordenador destaca por su alto nivel de construcción y su interesante software para el manejo del código máquina

El Memotech MTX 512 posee una serie de características que lo acercan en gran medida a lo que se denomina un ordenador "MSX standard" (véase p. 252); y si no fuera por la utilización del chip de sonido 76489 de Texas Instruments (MSX designa un AY-3-8910 de General Instruments), el MTX 512 podría considerarse como una de estas máquinas "estándar". Por otra parte, sin embargo, se ajusta a las especificaciones MSX al contar con una unidad central de proceso Z80, con un videoprocesador TMS9918/9928 de Texas Instruments, y una versión de BASIC que se halla aceptablemente cercana a la versión Microsoft.

El Memotech MTX 512 es una máquina tan completa y con un diseño tan elegante que no cabe la menor duda de que conseguirá numerosos adeptos. Su apariencia externa representa un avance considerable con respecto a otros muchos ordenadores, en los cuales, con bastante frecuencia, se introducen apretadamente componentes electrónicos de una complejidad muy elevada en una envoltura barata y muy poco dimensionada, de color negro y con un diseño atractivo, construida de aluminio.

La máquina está diseñada de forma que permita un fácil acceso al interior (simplemente quitando dos tornillos Allen y girando la cubierta inferior) para dejar a la vista el tablero de circuitos. Comparado con otros ordenadores, el MTX 512 tiene un número relativamente grande de chips. Es evidente que los diseñadores de la máquina prefirieron, o encontraron que era más económico, no emplear algunas ULA de gran capacidad. Mediante la utilización de una distribución más tradicional, consistente en numerosos chips montados de modo hermético, la máquina facilita un diagnóstico de fallos más rápido y fácil. En cambio, con la utilización de ULA, las averías son más difíciles de localizar e imposibles de reparar.

El manual de instrucciones no es tan bueno como el de otras marcas y, aparte de las cubiertas, no tiene ningún esquema en color. Otra desventaja la constituye la falta de un índice, lo cual dificulta su manejo. Sin embargo, es un manual bastante amplio y completo. Memotech ha decidido hacer una máquina "abierta", en el sentido de que no oculta ningún secreto al comprador. La información sobre la máquina es presentada con todo tipo de detalles: todos los esquemas de la memoria, tablas de posiciones útiles, formas de input/output, el diagrama de los circuitos y una buena introducción al lenguaje BASIC. También se incluyen unos capítulos específicos sobre NODDY (véase recuadro), el ensamblador-desensamblador y gráficos.

Una característica que distingue al Memotech MTX 512 de otros ordenadores es poseer un ensamblador-desensamblador que puede proporcionar, junto con el paquete de software Front Panel que se suministra, una programación de código máquina. El paquete ensamblador, sin embargo, no puede manejar direcciones simbólicas ni etiquetas; pero dado que, durante la programación, se mantienen notas pormenorizadas, resulta bastante adecuado para programas de tamaño medio. En próximos capítulos del curso veremos con mayor detalle tanto los paquetes ensambladores como el código máquina.

El Front Panel es una adición original en una máquina de este nivel, y es capaz de corregir cualquier tipo de error de código máquina. Desafortunada-

Teclado
Este teclado se encuentra entre los mejores que se montan en los ordenadores personales. Tiene 79 teclas del tipo máquina de escribir profesional, las cuales, en su parte posterior, tienen una hoja de acero. Esto hace que sean muy rígidas, lo que, junto con la fundición de aluminio, proporciona a la máquina una gran solidez

Interface cassette

Conectores de palancas de mando
Dispone de dos conexiones, que pueden adaptarse a cualquier palanca que emplee la norma Atari

Ampliación
El Memotech MTX 512 tiene la posibilidad de ser ampliado considerablemente. La primera adición importante sería un tablero de expansión de memoria y otro tablero de interface en serie, mediante dos conexiones RS232. Estas ampliaciones pueden emplearse para comunicaciones en serie normales o, con un software adecuado, en una red, lo cual hará que la máquina pueda competir en el campo de la educación

Chip reloj
El Z80 CTC suministra todas las funciones de control de tiempo usadas por el microprocesador

RAM usuario
El MTX 512 tiene 64 Kbytes estándar. El MTX 500 tiene 32 Kbytes

CPU
El microprocesador Z80 se utiliza con una velocidad de reloj de 4 MHz

Interface en paralelo
Esta conexión se ajusta a las normas Centronics para interfaces en paralelo y, junto con las interfaces RS232, permite que el MTX 512 dirija prácticamente cualquier impresora



Conector hi-fi
Esta interface, raramente montada en los ordenadores personales, permite la salida a través de un altavoz de hi-fi, que proporciona un sonido de mayor calidad

Chip de gráficos
El TMS 9928 de Texas Instruments controla todos los aspectos de la generación de video y hace que el MTX produzca unos gráficos de características similares al TI99/4A y al Sord M5. Asimismo, el sistema de funcionamiento de este ordenador presenta otras propiedades útiles, como, por ejemplo, la capacidad de dividir la pantalla en varias ventanas

MEMOTECH MTX 512

DIMENSIONES

488×202×56 mm

CPU

Z80

VELOCIDAD DEL RELOJ

4 MHz

MEMORIA

ROM: 24 K
RAM: 64 K de RAM usuario, más 16 K RAM video
Ampliable a 512 K

VISUALIZACION EN VIDEO

24 líneas de 40 caracteres, 16 colores con fondo y primer plano ajustables independientemente. 127 caracteres predefinidos y otros 127 definibles por el usuario

INTERFACES

Cassette, TV, monitor video compuesto

LENGUAJE SUMINISTRADO

BASIC, NODDY, ensamblador

OTROS LENGUAJES DISPONIBLES

No determinados

VIENE CON

Instalación y manuales BASIC, cable TV

TECLADO

79 teclas de alta calidad

DOCUMENTACION

Razonablemente completa, pero de presentación poco atractiva. Contiene información suficiente sobre el funcionamiento interno de la máquina, lo que permite a la mayoría de los programadores experimentados conseguir un control total

mente, éste es un aspecto de la máquina que posee una documentación muy somera, y aunque se proporciona una lista de las diversas órdenes, se facilita muy poca información sobre sus funciones, y muy pocos ejemplos de su empleo.

El Memotech MTX 512 puede ampliarse considerablemente, y con las diversas extensiones que están previstas, puede convertirse en una máquina muy capaz. Sin lugar a dudas, satisfará a numerosos usuarios y estimulará el desarrollo de un amplio software complementario.

NODDY

El sistema de software incluye un subconjunto de lenguaje NODDY y proporciona a la máquina una nueva dimensión. El NODDY, creado para ser utilizado por usuarios inexpertos, da la impresión de ser un lenguaje sencillo. Sin embargo, posee algunas órdenes de una elevada sofisticación. Está limitado porque tiene sólo 11 órdenes; además, no hay posibilidad de efectuar operaciones aritméticas. Ello es debido a que el lenguaje está ideado principalmente para manejar información textual. Quienes se inician en el manejo de ordenadores, a menudo hallan más fácil usar como datos básicos un texto que números

Sonido espectacular

El Oric-1 permite un sofisticado control del sonido

MUSIC y PLAY
El siguiente programa emplea las órdenes MUSIC y PLAY para tocar siete veces el acorde de *do mayor* invertido (*do, sol* y *mi*), usando cada vez una envoltura:

```
10 REM********
20 REM *ACORDE*
30 REM********
40 MUSIC 1,4,1,0:REM
   *DO*
50 MUSIC 2,3,8,0:REM
   *SOL*
60 MUSIC 3,3,5,0:REM
   *MI*
70 FOR E=1TO7:REM
   *SELEC.ENV.*
80 PLAY 7,0,E,750:REM
   *TOCA ACORDE*
90 PLAY 0,0,0,0:REM
   *STOP ACORDE*
100 WAIT 50:REM
    *PAUSA*
110 NEXT E:REM *OTRA
    ENV*
```

La gama de posibilidades del ordenador Oric-1 es muy amplia, y entre éstas quizá una de las más sobresalientes sea su capacidad de creación de sonido. Tiene una escala de siete octavas y en su versión estándar está provisto de tres osciladores, un generador de ruidos y siete envolturas preseleccionadas (véase p. 276) que pueden elegirse para dar forma a los sonidos producidos. La salida del sonido es a través del altavoz del televisor.

El BASIC del Oric-1 define un conjunto de órdenes para la elaboración del sonido: ZAP, PING, SHOOT y EXPLODE. El programa siguiente muestra cómo se pueden emplear estas órdenes, y da información sobre la orden WAIT, que detiene el ordenador durante un tiempo establecido, en centésimas de segundo (en este caso, dos segundos):

```
10 ZAP:WAIT 200
20 EXPLODE
30 GOTO 10
```

La orden SOUND se emplea sobre todo para producir efectos especiales, y se construye como sigue:

```
SOUND C,P,V
```

donde C=número del canal u oscilador (1-6); P=tono (10-5000), y V=volumen (0-15). Si el canal se coloca en 1, 2 o 3, se selecciona uno de los tres osciladores (4, 5 o 6 son equivalentes, pero también seleccionan el ruido). El tono no es del todo preciso; 10 es la nota más alta (a aproximadamente 10 KHz), y 5000 la más baja (a 100 Hz). El volumen máximo se obtiene colocando el mando a 15, pero el más agradable es a 6. Si V se sitúa en 0, el control es efectuado por una envoltura de volumen, seleccionada por la orden PLAY.

El inconveniente más importante de la orden SOUND es que no se puede determinar la duración de una nota. Además, SOUND no puede pararse por sí misma. El sonido de una nota sólo se detiene con el empleo de la orden PLAY y a continuación colocar todo ceros para que cese PLAY.

La orden MUSIC es ideal para especificar con precisión las notas. Su construcción sencilla hace que pueda entenderse con facilidad un programa de música relativamente complejo. El formato es:

```
MUSIC C, O, N, V
```

donde C=canal (1, 2 o 3); O=octava (0-6); N=nota (1-12), y V=volumen (0-15). Esta orden funciona de forma semejante a SOUND. El canal selecciona los osciladores 1, 2 o 3 (aunque en MUSIC no podemos obtener ruido). La gama de volumen abarca desde 0 (en cuyo caso el control lo efectúa la orden PLAY) hasta 15. La octava permite la selección de una octava determinada, en la que la nota (N) será parte de las órdenes. Una octava situada en 0 produce las notas más bajas, empezando en 32,7 Hz. Colocada en 6 se alcanzará la frecuencia de 3951,07 KHz. Para la nota (N) componente de la orden, los números 1 a 12 corresponden a las notas musicales normales, con la siguiente distribución:

1	2	3	4	5	6
DO	DO#	RE	RE#	MI	FA
7	8	9	10	11	12
FA#	SOL	SOL#	LA	LA#	SI

Luz y color

Gráficos del Spectrum: limitados pero fáciles de utilizar

El ordenador Spectrum representa un buen punto de partida para aquellos usuarios que estén interesados en gráficos en color de alta resolución. Su simplicidad de uso hace que el diseño de gráficos sea fácilmente accesible, incluso para quienes tengan una experiencia limitada en programación.

En este ordenador se dispone de los juegos normales de caracteres en mayúsculas y minúsculas, junto con varios caracteres exclusivos de Sinclair. Éstos pueden imprimirse en cualquiera de los ocho colores. Los colores para los caracteres, pantalla y recuadros se obtienen a través de las órdenes INK, PAPER y BORDER, respectivamente. Además del juego de caracteres estándar, el usuario puede definir hasta un total de 21 caracteres gráficos.

La visualización en pantalla consiste en 24 filas de 32 espacios cada una de ellas. Las dos últimas filas de la parte inferior, sin embargo, están reservadas para mensajes desde el ordenador o para entradas a través del teclado. Por tanto, la pantalla que en realidad se puede emplear es de 22×32 caracteres. En alta resolución, esto se convierte en 176×256 pixels. Una característica extremadamente útil del Spectrum es que tiene la posibilidad de mezclar en la pantalla gráficos de alta resolución junto con texto, permitiendo la creación de diagramas con epígrafes, gráficos, etc. Una vez que se ha diseñado una visualización en pantalla, es posible guardarla, mediante la orden SAVE, en una cinta magnética y volverla a utilizar cuando sea necesario. La orden SCREEN$ se encarga de efectuar este cometido, y puede usarse también para transferir el contenido de la pantalla a una impresora.



Para tocar la nota *la* a 440 Hz en el canal 1 a un volumen de 6, la orden debería ser:

MUSIC 1, 3, 10, 6

Sin embargo, para aprovechar todas las posibilidades del Oric-1, es mejor utilizar la orden MUSIC conjuntamente con PLAY. Esta última se forma de la manera siguiente:

PLAY C, N, E P

donde C=canal (0-7); N=ruido (0-7); E=envoltura (1-7); y P=período de envoltura (0-32767). Canal y ruido seleccionan unas opciones más complejas que las órdenes anteriores, tal como se indica a continuación:

Número	Canal	Ruido
0	Ningún osc.	Ninguno
1	Osc. 1	+Osc. 1
2	Osc. 2	+Osc. 2
3	Oscs. 1 y 2	+Oscs. 1 o 2
4	Osc. 3	+Osc. 3
5	Oscs. 1 y 3	+Oscs. 1 o 3
6	Oscs. 2 y 3	+Oscs. 2 o 3
7	Oscs. 1, 2 y 3	+Oscs. 1, 2 o 3

Las órdenes de MUSIC (o SOUND) definidas previamente, con el volumen situado en 0, pueden darse (PLAY) conjuntamente, de acuerdo con el número de canal seleccionado, para producir acordes de hasta tres notas. La parte correspondiente al ruido de esa orden selecciona cuál de los osciladores, en el caso de que haya alguno, es el que debe efectuar la mezcla. La E selecciona una de las siete envolturas de volumen para la nota o notas especificadas. Estas opciones están indicadas en el manual de instrucciones del Oric.

El único control variable sobre la envoltura es P, que permite al programador especificar su duración total (desde 0 hasta 32767). El período varía con cada envoltura, pero como dato orientativo se puede decir que con un valor de 5000 dura, aproximadamente, 2 segundos.

Las órdenes de sonido del Oric-1 son fáciles de usar, y constituyen una prueba evidente del esfuerzo realizado para incorporar a este ordenador unas configuraciones de alto nivel. Sólo hay otro ordenador personal que posee también un BASIC práctico y un control sobre la envoltura: es el BBC Micro, cuyas posibilidades de creación de sonido son mucho más notables. A pesar de ello, el bajo coste del Oric-1 hace que éste sea un ordenador valioso para quienes quieran componer música electrónica con un presupuesto bajo.

SOUND
Este pequeño programa utiliza la orden SOUND para producir un ruido como de una nave espacial que aterriza:

```
10 REM**********
20 REM *ATERRIZAJE*
30 REM**********
40 FOR P=10 TO 3000
   STEP 10
50 SOUND 2,P,6
60 PLAY 2,0,1,1
70 NEXT P
80 WAIT 75
90 PLAY 0,0,0,0
100 END
```

Asimismo puede lograrse que en la pantalla aparezca un gráfico de baja resolución. Para ello debe usarse la orden PRINT AT, la cual permite situar el carácter tanto horizontal como verticalmente. Este ordenador dispone también de una serie de efectos especiales. Además de la visualización, los caracteres pueden ser intermitentes o fijos empleando las órdenes FLASH o BRIGHT respectivamente. La orden OVER es también útil para la baja resolución. Permite que un segundo carácter se combine con el original cualquiera que sea su posición. Esto resulta particularmente efectivo para mezclar texto y gráficos de alta resolución, puesto que resulta posible escribir sobre los diagramas sin borrarlos. Sin embargo, este efecto debe emplearse con cierta precaución, pues si se recompone el color mediante la orden INK en un cuadrado particular, la visualización original también cambia al color nuevo.

La visualización en pantalla está gobernada por dos áreas de la memoria: una de ellas representa en la pantalla los caracteres, y la otra mantiene la información correspondiente a los atributos de una posición específica del carácter. Esta información se refiere a los siguientes puntos: los colores de los caracteres (INK) y de la pantalla (PAPER); si el carácter es intermitente (FLASH), etc. Estos atributos están representados por un solo byte, y mediante la orden ATTR de un programa en BASIC puede determinarse el estado de cualquier posición de pantalla.

Mediante las órdenes en BASIC, el ordenador Spectrum logra fácilmente visualizaciones de alta resolución. Esto se debe, en gran parte, a que no existe una pantalla independiente para alta resolución, haciendo que sea sencillo mezclar gráficos y texto en una sola representación.

Las órdenes en BASIC incluyen:

PLOTx,y

Esto hace corresponder el pixel de coordenadas (x,y) con el color que tenga el carácter (INK).

DRAWx,y,p

DRAW crea una línea entre la posición real del cursor y las coordenadas especificadas en la orden. Si se añade un tercer número, la línea dibujada será un arco de circunferencia. Este tercer número normalmente será una fracción de π (3,14159...). Si este tercer dígito fuera π, se obtendría la representación de una semicircunferencia, mientras que con $\pi/2$ trazaría un cuadrante. La orientación de la concavidad o convexidad del arco se determina haciendo que el tercer número sea positivo o negativo.

CIRCLEx,y,r

La orden CIRCLE dibuja un círculo, de centro (x,y) y radio r. Con numerosas órdenes CIRCLE en BASIC es posible formar elipses mediante la deformación de la circunferencia primitiva. Sin embargo, el Spectrum no posee esta propiedad.

Existe una desventaja importante en el uso del color en gráficos de alta resolución. Como consecuencia de la posibilidad de mezclar texto y diagramas, sólo puede especificarse un color de carácter (INK) en cada cuadrado de ocho por ocho pixels. Así, si se cruzan dos líneas de color diferente, en el interior del cuadrado donde se produce la intersección, todos los pixels tomarán el último color (INK).

Sonría, por favor
Este programa muestra el uso de las órdenes de alta resolución PLOT, CIRCLE y DRAW, para crear una cara "sonriente":

```
10 REM *CARA
   SONRIENTE*
20 CLS
30 BORDER 6
40 PAPER 6
50 INK 2
60 CIRCLE 122,88,50
70 CIRCLE 97,108,5
80 CIRCLE 147,108,5
90 PLOT 92,68
100 DRAW 152,68,PI/3
110 END
```

Control centralizado

En un ordenador personal pueden conectarse sensores para luz, temperatura y otros efectos. La información puede usarse para el control de un sistema de calefacción y una alarma antirrobo

Los microprocesadores son utilizados cada día en mayor número en una amplia gama de aplicaciones domésticas, como, por ejemplo, lavadoras automáticas, lavavajillas, grabadoras de video y unidades de calefacción central. No resulta sorprendente, por consiguiente, que se puedan interconectar estos chips de control de forma que los aparatos de toda la casa puedan compartir entre ellos la información, o transmitir los datos a un sistema central de control. Es perfectamente posible diseñar y construir sistemas de control centralizado que regulen las aplicaciones domésticas. Estos sistemas pueden dividirse en tres categorías: sistemas permanentes y temporales y controladores interconectados.

Los pertenecientes a la primera categoría están comercializados, pero también cabe la posibilidad de diseñarlos uno mismo. Éstos se acoplan a un ordenador personal convencional y emplean interfaces específicas para conectarse directamente, y regulan unidades eléctricas o electromecánicas como luces o termostatos. Sin embargo, para montar un sistema de este tipo, es necesario poseer un conocimiento amplio del hardware del ordenador y ser capaz de escribir los programas de control necesarios. Los sistemas permanentes tienen también la limitación fundamental de que el programa debe funcionar continuamente. Cualquier interrupción en el suministro del fluido eléctrico hará que el dispositivo quede, en el mejor de los casos, bloqueado en una posición determinada y no pueda llevar a cabo los ajustes y órdenes indicados por el programa de control.

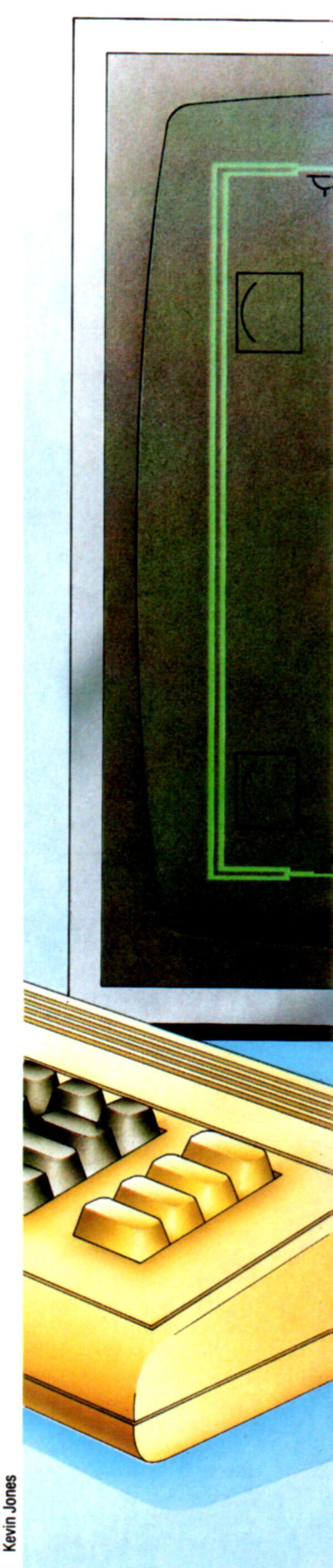
La segunda categoría de sistemas de control temporal emplea señales electrónicas generadas en circunstancias determinadas por los dispositivos de regulación conectados a la calefacción central, alarma antirrobo, o detectores de fuego y humo. Cuando alguno de estos aparatos envía alguna información fuera de lo normal al ordenador, éste emite una señal prioritaria que hace que quede interrumpido el programa en uso. Si bien un sistema de este tipo debe funcionar continuamente, es mucho más adecuado para hacer frente a una avería en la fuente de alimentación, puesto que los dispositivos que controla son parcialmente autorregulables.

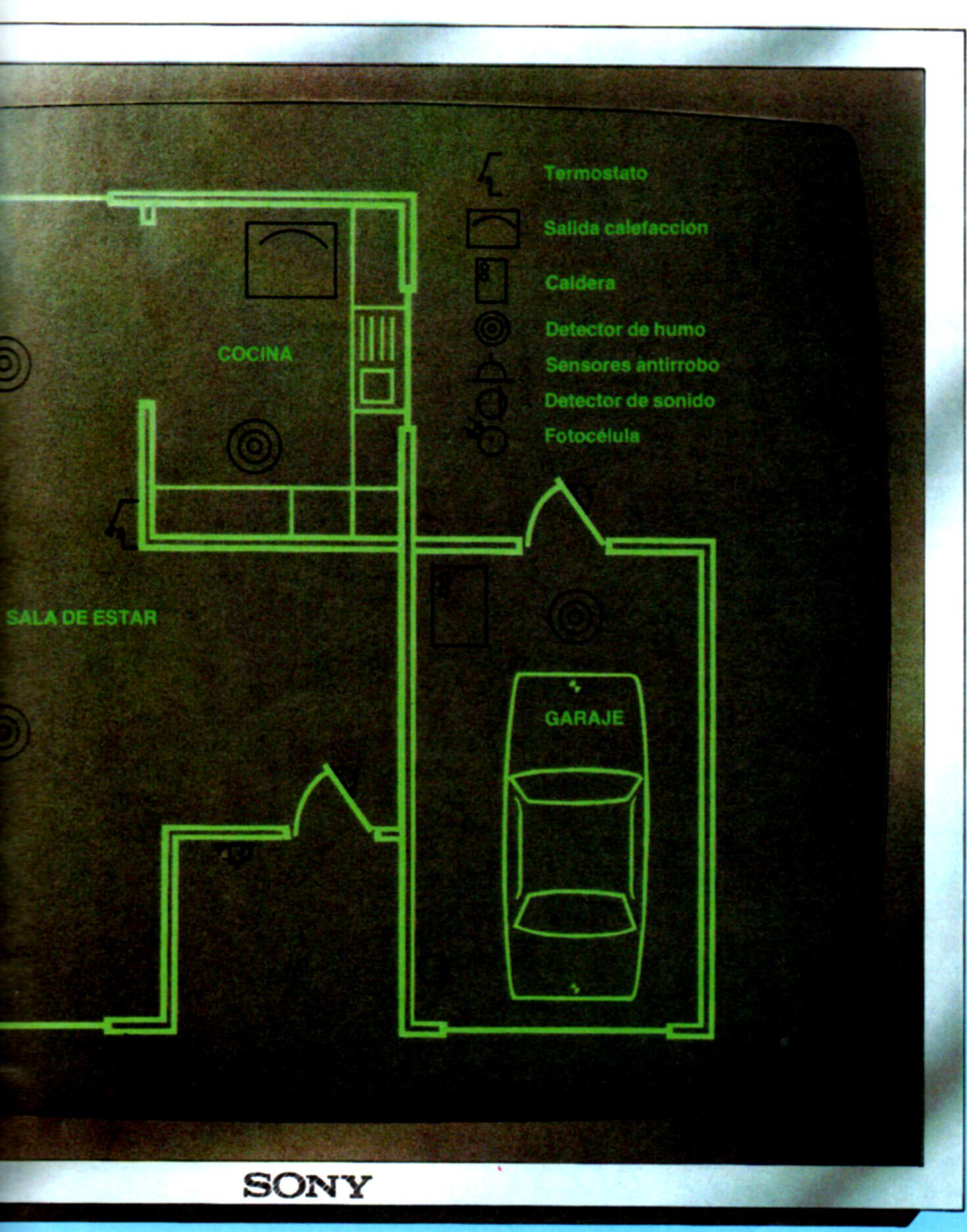
La memoria del ordenador contiene varios programas: uno para cada dispositivo conectado a él, además del programa que se está utilizando. Supongamos que en un momento determinado el ordenador está ocupado con un juego de aventuras y que entonces entra en funcionamiento el detector de humo. En respuesta a la señal, el ordenador interrumpe el programa del juego (manteniendo toda la información necesaria sobre la situación del juego en aquel preciso momento), y empieza a funcionar el programa detector de humo. En la pantalla del ordenador aparecerá un mensaje comunicando que se ha detectado un fuego potencial; o, si no se estaba empleando el ordenador, puede empezar a sonar una alarma. Una vez que se ha detectado el origen del humo y se ha solucionado el problema, se puede volver al punto exacto del juego en el momento en que se había interrumpido. Sin embargo, si la señal hubiera procedido del regulador del sistema de calefacción central, el ordenador verificaría el tiempo y los sensores de temperaturas externa e interna y, si fuera necesario, pondría en funcionamiento la caldera, realizando todos estos pasos con una rapidez tal que no se podría detectar que el juego hubiera sufrido una interrupción.

Un ejemplo de la tercera categoría (controladores interconectados) es un sistema llamado BSR Home Controller. Este ingenioso dispositivo utiliza el tendido eléctrico normal de una casa para controlar las unidades enchufadas a cualquier conexión de un circuito determinado. Cada controlador es identificado por un número de código (en realidad, una dirección), que le permite ser conectado o desconectado mediante una señal de alta frecuencia a través del tendido eléctrico normal. Sin embargo, el montaje de este tipo de interfaces es muy peligroso. Únicamente un técnico cualificado debe efectuar las conexiones desde las salidas del ordenador a la red de corriente.

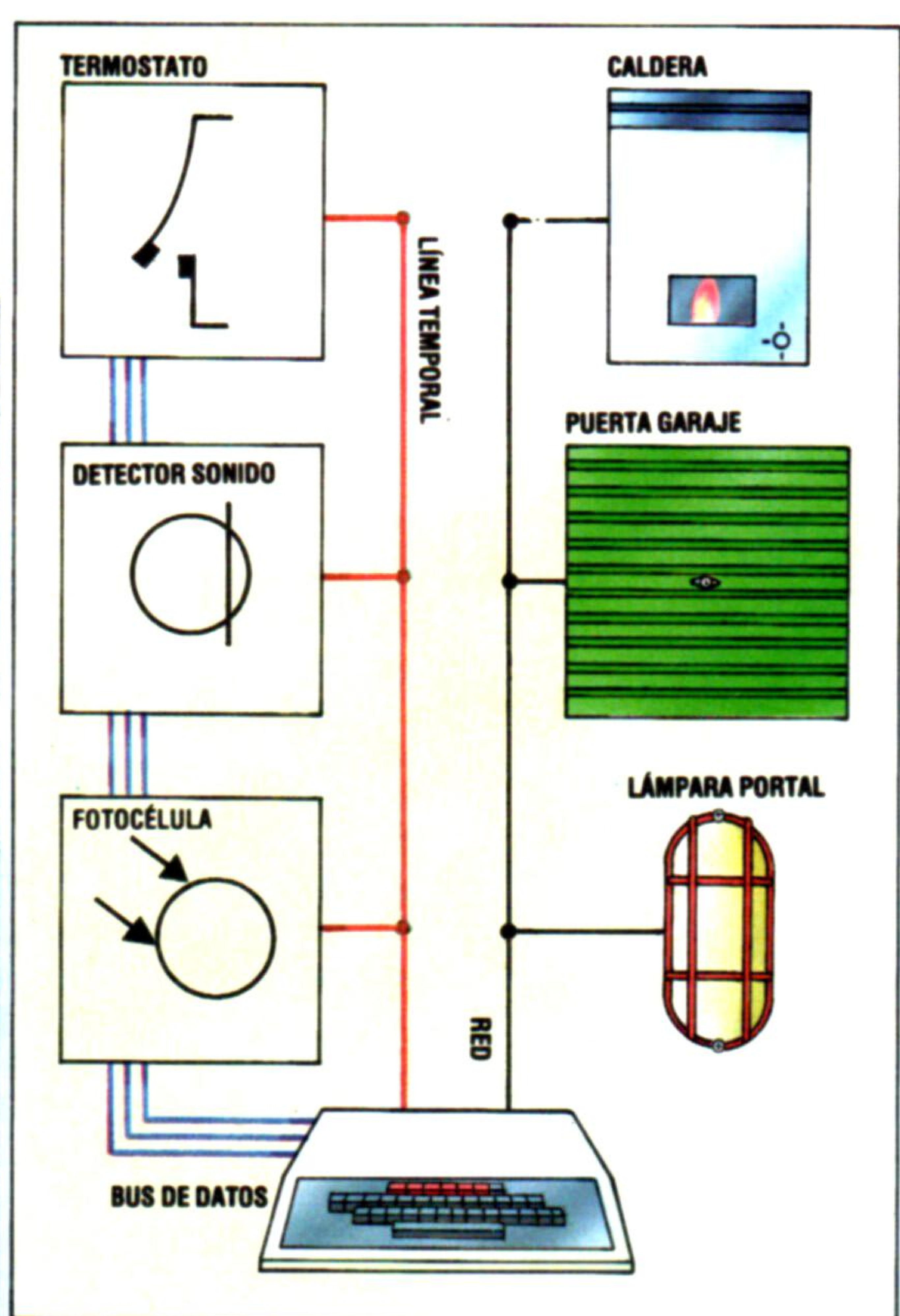
Una vez que haya sido instalado el sistema de control, el siguiente paso consistirá en que éste sea capaz de funcionar mediante algún tipo de control remoto; de esta forma será posible que, incluso hallándose el usuario fuera de casa, pueda desconectar la calefacción o poner en funcionamiento la alarma antirrobo. En cualquiera de estos sistemas puede montarse un dispositivo de comunicaciones estándar, por ejemplo un modem, que permite realizar el control mediante una terminal a distancia. Cuando el sistema disponga de esta característica, para tener acceso a él será necesario utilizar algún tipo de contraseña.

El coste principal de montaje de un sistema de control computerizado en una casa lo comporta la compra del hardware para conectar el ordenador a la red de suministro eléctrico, ya que es necesario montar numerosos aislantes, relés e interruptores de estado sólido para poder hacer el montaje de una forma segura y eficiente. Pero, probablemente, la tarea más absorbente para el usuario de un ordenador personal, para poner en funcionamiento un sistema de este tipo, sea la escritura del software. Debido a que estos dispositivos se fundamentan en su rapidez de respuesta (de nada serviría que la alarma contra fuego empezara a funcionar después de que la casa hubiera ardido), los programas de control deben escribirse en código máquina. Este tipo de programas aún no están comercializados, pero en un futuro próximo será posible adquirirlos.


Kevin Jones







Dirección doméstica
Este diagrama muestra dos de las técnicas mediante las cuales un ordenador personal puede dirigir cierto número de aplicaciones domésticas. Cuando uno de los tres sensores de la izquierda tiene algo que transmitir, envía un pulso electrónico a través del circuito común. Éste conduce al microprocesador, el cual interrumpe provisionalmente cualquier programa que esté en funcionamiento y salta a una rutina especial, que lee toda la información que el sensor suministra al bus de datos. Los componentes de la derecha están conectados a una red y, por lo tanto, el ordenador puede activar cualquiera de ellos simplemente enviando un paquete de datos consistente en, por ejemplo, el número del dispositivo de la puerta del garaje y la instrucción para abrirla

Centro de control

Es posible obtener una representación esquematizada de su casa en la pantalla del ordenador. Los sistemas de seguridad y control computerizados de plantas industriales emplean dichos métodos. Desde luego, si el ordenador funciona con el método "temporal", no es necesario obtener una visualización en la pantalla, puesto que el software efectúa todas las operaciones de control internamente, por lo general sin un retraso perceptible en el programa que funciona en ese momento en el ordenador. Es muy probable que no pasen muchos años antes de que las casas sean diseñadas incorporando dispositivos de este tipo

Tiempo y movimiento

En BASIC, la clasificación en serie puede ser una operación que ocupe tiempo, pero facilita la búsqueda de registros específicos

A estas alturas del curso de programación, ya hemos desarrollado la mayor parte de la codificación necesaria para crear las entradas en nuestra agenda de direcciones en "base de datos", pero aún no hemos abordado la programación necesaria para conservar las entradas en cinta magnética o disco. El único aspecto importante que no hemos tratado es la formación de una rutina adecuada para crear el campo MODCAM$.

El programa completo para ello se incluye en el presente capítulo. Primero, todos los caracteres deben ser convertidos al tipo de caja alta (letras mayúsculas) entre las líneas 10250 y 10330. Luego, de la línea 10350 a la 10370 se cuentan los caracteres de la serie y se verifican uno por uno para comprobar si existe un espacio. El último espacio encontrado ajusta la variable S al valor correspondiente a su posición en la serie.

De la línea 10400 a la 10420 se traspasan los caracteres, uno a uno, desde la serie en que figuran en mayúsculas hasta CNOM$. Los caracteres son trasferidos, hasta que se llega al último espacio, si tienen un valor ASCII mayor que 64. Cualquier carácter que no supere esta prueba es ignorado. Así, el proceso elimina puntos (ASCII 46), apóstrofos (ASCII 39), espacios (ASCII 32) y los demás signos de puntuación. Desde la línea 10450 a la 10470 se procede igual para los caracteres situados tras el espacio final, traspasándolos a SNOM$.

Si N$ contiene únicamente una palabra, SUPERMERCADO, por ejemplo, la variable S será 0 y todos los caracteres serán transferidos a SNOM$. La variable utilizada para designar los nombres de pila ha sido llamada CNOM$, mientras que la variable SNOM$ se utiliza para designar el apellido.

Las líneas 10490 y 10500 son necesarias para convertir en cero las variables alfanuméricas empleadas en esta rutina, antes de que sean usadas otra vez. Éste es un punto que no hay que perder de vista siempre que se empleen estructuras del tipo LET X$=X$+Y$. Si se produce un fallo en la "limpieza" de las variables, se originará una acumulación cada vez mayor de caracteres erróneos en las variables siempre que sean utilizadas. Téngase en cuenta que OPCN es situado en 0 en la rutina INCREG, puesto que únicamente se quiere comprobar que el usuario añade un registro si no hay ninguno en el archivo (es decir, la primera vez que se utiliza el programa).

Ahora que ya se dispone de un medio de añadir tantos registros como se desee en el archivo, es necesario imaginar cómo se puede conservar el archivo en cinta magnética o disco. La forma más sencilla consistiría en escribir todos los registros en el archivo de datos (DAT.AGCO, en esta versión del programa) en el orden en que han sido introducidos. La desventaja más importante de este método de aproximación se pone de manifiesto cuando es necesario buscar en el archivo un registro determinado. Si no se tiene la entera seguridad de que todos los registros del archivo están ordenados en alguna forma, la única manera de hallar un registro sería examinando cada registro desde el principio con el fin de comprobar si existe la correspondencia adecuada. Si el registro que se está buscando da la casualidad que era el último que se introdujo, será necesario examinar cada uno de los registros de la base de datos antes de poder localizar el que se buscaba. Si el último registro introducido hubiera sido el correspondiente a Lorenzo Díaz (es decir, MODCAM$ (TAMA−1)="DIAZ LORENZO"), una rutina de búsqueda haría que el registro estuviera en algún punto cercano al principio del archivo (si los registros estuvieran ordenados). Desafortunadamente, tanto la tarea de búsqueda como la de ordenación son actividades que consumen mucho tiempo; por consiguiente, hay que determinar las prioridades y considerar qué resulta más interesante. Aquí se ha adoptado el principio de que una agenda de direcciones es con mayor frecuencia consultada que modificada (añadiendo nuevas direcciones o en alguna otra forma). En tal caso, es mejor suponer que habrá más búsquedas que ordenaciones, y habrá que asegurarse de que los registros están ordenados, antes de ser almacenados en el archivo de datos, una vez utilizado el programa.

Sin perder de vista esta observación, se crea una variable, llamada RMOD, para ser usada como bandera. Puede tener dos valores: 0 o 1. Inicialmente toma el valor 0, para indicar que ningún registro ha sido modificado durante la actual ejecución del programa. Cualquier operación que modifique el archivo en alguna forma (por ejemplo, añadiendo un nuevo registro) sitúa RMOD en 1. Las operaciones que "necesiten saber" si ha sido modificado el archivo, verificarán el valor de RMOD antes de empezar a procesar. Por ejemplo, SAPROG, la rutina que guarda el archivo y las salidas del programa, comprueba RMOD, en la línea 11050. Si RMOD=0, no es necesario ningún tipo de clasificación o conservación, porque se da por supuesto que el archivo de datos, en cinta magnética o en disco, está completamente ordenado y sin modificar. Otras rutinas, tales como las que buscan un registro determinado a través del archivo, también necesitan verificar el valor de RMOD. Si éste corresponde a 0, la búsqueda (u otra operación) puede continuar. Si RMOD es 1, lo primero que deberá hacer la rutina será llamar a la rutina de clasificación. Efectuada la clasificación de todo el archivo, la rutina de clasificación volverá a poner 0 en RMOD.

En nuestro caso, la rutina de clasificación, llamada *CLSREG* en el listado del programa, vuelve a colocar la variable RMOD en 0 en la línea 11320, una vez que todos los registros han sido clasificados. Antes de continuar y centrar la atención en *SAPROG* (la rutina que guarda el archivo en cinta magnética o disco y luego da por terminado el pro-

grama), digamos algunas palabras sobre qué se entiende por *CLSREG*. Consiste en una técnica de clasificación sencilla, denominada "clasificación burbuja" (véase p. 286). Existen muchas formas de clasificar datos, y ésta es una de las más simples y lentas. Hay otras rutinas más eficaces, pero los tipos de clasificaciones más sofisticadas resultan mucho más difíciles de entender que la mencionada anteriormente. El tipo de rutina a emplear depende del número de datos que haya que clasificar. La "complejidad de tiempo" de una clasificación burbuja como la nuestra es n^2. En otras palabras, el tiempo necesario para clasificar los datos se incrementa con el cuadrado del número de elementos a ordenar. Si dos elementos requirieran cuatro milisegundos para ser clasificados, cuatro necesitarían 16 milisegundos, 50 elementos dos segundos y medio y si fueran 1 000 se tardaría más de 15 minutos. Una espera de dos o tres segundos es aceptable en un programa semejante al nuestro, pero de ninguna manera una demora de un cuarto de hora.

La forma en que ha sido escrito este programa permite un máximo de 50 registros únicamente, por tanto no se presentarán problemas de espera.

Los datos que se están clasificando son las series de caracteres de MODCAM$(L) y MODCAM$(L+1). Los registros únicamente son intercambiados si MODCAM$(L) es mayor que MODCAM$(L+1), y el campo de índice (que por el momento no se está utilizando) es actualizado en las líneas 11490 y 11570. Cada vez que se intercambian dos registros, la variable S (para indicar que se ha producido un intercambio) toma el valor 1. Cuando la rutina de clasificación alcanza la línea 11290, verifica el valor de S y vuelve atrás para comparar otra vez todos los registros. Cuando todos ellos estén en orden, el valor de S se situará en 0 y la rutina habrá finalizado, una vez que el valor de RMOD sea también 0.

La rutina SAPROG (denominada *SAPROG* en el listado del programa) empieza en la línea 11000. El primer paso que efectúa es verificar si, durante la ejecución actual del programa, se ha modificado algún registro (línea 11050: IF RMOD=0 THEN RETURN). Si no se ha producido ninguna modificación en el archivo, no será necesario proceder otra vez a la grabación en cinta magnética o disco y la rutina volverá al programa principal. Esto hace que volvamos de nuevo a la línea 100, la cual verifica el valor de OPCN. Si tiene un valor 9 (tal como debería ser si *SAPROG* está siendo procesado), el programa principal sigue hasta la orden END, en la línea 110.

Si el programa encuentra que RMOD tiene el valor 1 en la línea 11050, significa que uno o más registros han sido modificados en alguna forma y que existe alguna posibilidad de que ya no estén en orden. Si se presenta esta situación, la rutina *SAPROG* hace entrar en funcionamiento la rutina de clasificación (línea 11070) y luego, tras ser clasificados todos los registros, los conserva en cinta magnética o disco.

La rutina de guardar (*GRDREG*) es llamada en la línea 11090 y comienza a actuar en la línea 12000. *GRDREG*, en el listado principal, está escrito en BASIC Microsoft, por eso es importante recordar que los detalles del manejo del archivo varían de una versión a otra de BASIC (véase "Complementos al BASIC"). La línea 12030 abre el archivo de datos DAT.AGCO y asigna el número de canal #1 a la operación. La línea 12050 establece los límites del bucle que cuenta todos los registros a través del archivo. El límite superior es TAMA-1, y no TAMA, porque esta última variable tiene siempre un valor mayor en una unidad al número de los registros vigentes en el archivo (así, si se añade un registro nuevo, no será escrito sobre uno ya existente).

El formato de las líneas 12060 y 12070 es particularmente digno de ser tenido en cuenta. Cada campo está separado por ",", que es también enviado al archivo. La mayoría de versiones de lenguaje BASIC necesitan esta coma, porque INPUT# y PRINT# trabajan de la misma forma que las órdenes INPUT y PRINT normales. Supóngase la sentencia INPUT X,Y,Z. Ésta espera que se produzca una entrada a través del teclado semejante a 10,12,15<CR>, la cual atribuiría los valores 10, 12 y 15 a X,Y, y Z respectivamente. Sin comas, la orden INPUT no sería capaz de decir dónde termina cada dato y asignaría toda la información a la primera variable. De una forma semejante, la orden INPUT# (en la mayoría de las versiones de BASIC) no podría indicar dónde termina cada registro de archivo de datos e intentaría llenar cada variable alfanumérica con tantos datos como pudiera. Puesto que la mayor parte de las variables alfanuméricas en BASIC pueden contener hasta 255 caracteres, los datos del archivo estarían todos pronto asignados, mucho antes de que hubiera terminado el bucle FOR L=1 TO TAMA-1. Esto ocasionaría un mensaje erróneo INPUT PAST END (indicando que se había emitido una orden INPUT después de agotar todos los datos), y las variables alfanuméricas (como, por ejemplo, NOMCAM$(x)) contendrían muchos más datos de lo debido.

Una vez que todos los registros han sido almacenados en el archivo de datos, desde L=1 TO TAMA-1, *GRDREG* RETURN vuelve a la línea 90 del programa principal. La línea 100 verifica el valor de OPCN para comprobar si la última operación era *SAPROG* o no. Si era 9 (guardar y salir), el programa continúa hasta la sentencia END, en la línea 110. Si OPCN es otro valor, el programa vuelve atrás hasta *ELECCN* y permite que el usuario elija de nuevo otra opción.

Por último, debemos mencionar la rutina *FINARCH*, que empieza en la línea 12500. Constituye una alternativa posible a la sentencia de la línea 1510. Tal como está configurado el programa, éste depende de la presencia de una función de final de archivo: IF EOF(1)=−1 THEN LET L=50. Todos los lenguajes BASIC tienen alguna forma de indicar que se ha alcanzado el final de un archivo, bien sea mediante una función especial como EOF(x) o un PEEK a una posición especial de la memoria. Se sugiere la rutina *FINARCH* de la línea 12500 si no se dispone de una función EOF, en cuyo caso la línea 1510 debería ser sustituida por GOSUB 12500.

Complementos al BASIC



Antes de procesar el programa de la agenda de direcciones, se debe crear, en una cinta magnética, el archivo de campo de nombres. El siguiente programa tiene esta finalidad.

```
10 REM PROGRAMA PARA CREAR
   ARCHIVO NCAM EN CINTA
20 DIM Z$(1,30)
30 LET Z$(1)="@ VACIO"
40 SAVE "NCAM" DATA Z$( )
50 STOP
```

SPECTRUM

Cuando se pare el programa, rebobinar la cinta y digitar VERIFY "NCAM" DATA Z$ ( ) para verificar que ha funcionado SAVE. Esta verificación de los datos y programas almacenados en cinta magnética duplica el tiempo de espera hasta que se para la grabadora de cassette, pero es una norma general que merece la pena tener en cuenta, antes de desconectar la máquina.
A continuación se indican las versiones para el ordenador Spectrum de las líneas y subrutinas del listado principal:

```
1100 REM *CREMAT*
1110 DIM N$(50,30)
1120 DIM M$(50,30)
1130 DIM C$(50,30)
1140 DIM D$(50,15)
1150 DIM P$(50,15)
1160 DIM T$(50,15)
1170 DIM X$(50,30)
1180 DIM B$(30)
1190 DIM Z$(30)
1250 LET Z$="@VACIO"
.
1400 REM *LEARCH* SR
1410 LOAD "NCAM" DATA N$( )
1420 IF N$(1)=Z$ THEN LET
     Q$=Z$:RETURN
1430 LOAD "MCAM" DATA M$( )
1440 LOAD "CCAM"DATA C$( )
1450 LOAD "DCAM" DATA D$( )
1460 LOAD "PCAM" DATA P$( )
1470 LOAD "TELCAM" DATA T$( )
1480 LOAD "INDCAM" DATA X$( )
1490 REM *FINARCH*
1500 GOSUB 12500
.
1540 RETURN
.
1640 IF Q$=Z$ THEN LET TAMA=1
.
3520 IF Q$=Z$ THEN GOSUB 3860:RETURN
.
3810 LET OPCN=CODE A$-48
.
10090 LET Q$=" "
.
10200 REM*MODNOM* SR
.
10250 LET R$=N$(TAMA):LET S$=""
10260 FOR L=1 TO LEN (R$)
10270 LET A$=R$(L)
10280 LET T=CODE A$
10290 IF T>=97 THEN LET T=T-32
10300 LET A$=CHR$ T
10310 LET S$=S$+A$
10320 NEXT L
10330 LET R$=S$:LET S$="":LET
      A$="":LET T=LEN(R$)
10340 REM LOCALIZAR ULTIMO ESPACIO
10350 FOR L=1 TO T
10360 IF R$(L)=" " THEN LET S=L:LET
      L=T
10370 NEXT L
10380 REM QUITAR SOBRANTE
10390 REM ALMACENAR NOMBRES DE
      PILA EN S$
10400 FOR=1 TO S-1
10410 IF CODE (R$(L))>64 THEN LET
      S$=S$+R$(L)
10420 NEXT L
10430 REM QUITAR SOBRANTES
10440 REM ALMACENAR APELLIDOS EN A$
10450 FOR L=S+1 TO LEN (R$)
10460 IF CODE(R$(L))>64 THEN LET
      A$=A$+R$(L)
10470 NEXT L
10480 LET M$(SIZE)=A$+" "+S$
10490 LET S$="":LET A$=""
10510 RETURN
```

N.B. Debido a la forma en que el Spectrum trata las series, la rutina anterior divide el nombre en el primer espacio, no en el último.

```
12000 REM *GRDREG* SR
12030 SAVE "NCAM" DATA N$( )
12040 SAVE "MCAM" DATA M$( )
12050 SAVE "CCAM" DATA C$( )
12060 SAVE "DCAM" DATA D$( )
12070 SAVE "PCAM" DATA P$( )
12080 SAVE "TELCAM" DATA T$( )
12090 SAVE "INDCAM" DATA X$( )
.
12150 RETURN
.
12500 REM *FINARCH* SR
12510 LET TAMA=50
12520 FOR L=1 TO 50
12530 IF N$(L)=B$ THEN LET
      TAMA=L:LET L=50
12540 NEXT L
12560 RETURN
```

LYNX

El ordenador Lynx no admite la orden STR$. Véase "Complementos al BASIC", p. 357. Para dimensionar matrices mediante la orden DIM, véase "Complementos al BASIC", p. 275.

EOF

En los ordenadores Commodore 64 y Vic-20, reemplazar la línea 1520 por:

```
1520 IF ST AND 64 THEN LET L=50
```

En el Dragon 32, suprimir la línea 1520 y sustituirla por:

```
1485 IF EOF(-1) THEN GOTO 1510
```

En el BBC Micro, reemplazarla por:

```
1520 IF EOF#X THEN LET L=50
```

donde X es la variable numérica empleada en la orden OPENOUT (véase p. 319).

OPEN CLOSE

Véase p. 319.

```
10 REM    'PROPRI'
20 REM    *INICIL*
30 GOSUB  1000
40 REM    *PRESEN*
50 GOSUB  3000
60 REM    *ELECCN*
70 GOSUB  3500
80 REM    *EJECUT*
90 GOSUB  4000
100 IF OPCN <> 9 THEN 60
110 END
1000 REM  SUBRUTINA *INICIL*
1010 GOSUB 1100: REM SUBRUTINA *CREMAT* (CREAR
     MATRICES)
1020 GOSUB 1400: REM SUBRUTINA *LEARCH* (LEER ARCHIVOS)
1030 GOSUB 1600: REM SUBRUTINA *ESBAND* (ESTABLECER
     BANDERAS)
1040 REM
1050 REM
1060 REM
1070 REM
1080 REM
1090 RETURN
1100 REM SUBRUTINA *CREMAT* (CREAR MATRICES)
1110 DIM NOMCAM$ (50)
1120 DIM MODCAM$ (50)
1130 DIM CLLCAM$ (50)
1140 DIM CIUCAM$ (50)
1150 DIM PROCAM$ (50)
1160 DIM TELCAM$ (50)
1170 DIM INDCAM$ (50)
1180 REM
1190 REM
1200 REM
1210 LET TAMA = 0
1220 LET RMOD = 0
1230 LET SVED = 0
1240 LET CURS = 0
1250 REM
1260 REM
1270 REM
1280 REM
1290 REM
1300 RETURN
1400 REM SUBRUTINA *LEARCH*
1410 OPEN "I",#1, "DAT.AGCO"
1420 INPUT #1, TEST$
1430 IF TEST$ = "@VACIO" THEN GOTO 1530: REM CERRAR Y
     RETORNAR
1440 LET NOMCAM$ (1) = TEST$
1450 INPUT #1,MODCAM$ (1), CLLCAM$ (1), CIUCAM$ (1),
     PROCAM$ (1), TELCAM$ (1)
1460 INPUT #1, INDCAM$ (1)
1470 LET TAMA = 2
```

```
1480 FOR L = 2 TO 50
1490 INPUT #1, NOMCAM$ (L), MODCAM$ (L), CLLCAM$ (L),
     CIUCAM$ (L), PROCAM$ (L)
1500 INPUT #1, TELCAM$ (L), INDCAM$ (L)
1510 LET TAMA = TAMA + 1
1520 IF EOF(1) = -1 THEN LET L = 50
1530 NEXT L
1540 CLOSE #1
1550 RETURN
1600 REM SUBRUTINA $ESBAND$
1610 REM ESTABLECE BANDERAS DESPUES DE $LEARCH$
1620 REM
1630 REM
1640 IF TEST$ = "@VACIO" THEN LET TAMA = 1
1650 REM
1660 REM
1670 REM
1680 REM
1690 RETURN
3000 REM SUBRUTINA $PRESEN$
3010 PRINT CHR$ (12):REM LIMPIAR PANTALLA
3020 PRINT
3030 PRINT
3040 PRINT
3050 PRINT
3060 PRINT TAB(11); "$BIEN VENIDO A LA$"
3070 PRINT TAB(9); "$AGENDA COMPUTERIZADA$"
3080 PRINT TAB(12); "$DE MI COMPUTER$"
3090 PRINT
3100 PRINT TAB(0); "(PULSE BARRA ESPACIADORA PARA CONTINUAR)
3110 FOR L = 1 TO 1
3120 IF INKEY$ <> " " THEN L = 0
3130 NEXT L
3140 PRINT CHR$(12)
3150 RETURN
3500 REM SUBRUTINA $ELECCN$
3510 REM
3520 IF TEST$ = "@VACIO" THEN GOSUB 3860: REM SUBRUTINA
     $PRIMERA$
3530 IF TEST$ = "@VACIO" THEN RETURN
3540 REM 'IMMENU'
3550 PRINT CHR$(12)
3560 PRINT "SELECCIONE UNO DE LOS SIGUIENTES"
3570 PRINT
3580 PRINT
3590 PRINT
3600 PRINT "1. HALLAR REGISTRO (DE NOMBRE)"
3610 PRINT "2. HALLAR NOMBRES (DE NOMBRE INCOMPLETO)"
3620 PRINT "3. HALLAR REGISTRO (DE CIUDAD)"
3630 PRINT "4. HALLAR REGISTRO (DE INICIAL)"
3640 PRINT "5. LISTAR TODOS LOS REGISTROS"
3650 PRINT "6. AGREGAR REGISTRO NUEVO"
3660 PRINT "7. MODIFICAR REGISTRO"
3670 PRINT "8. BORRAR REGISTRO"
3680 PRINT "9. SALIR Y GUARDAR"
3690 PRINT
3700 PRINT
3710 REM 'ASOPCN'
3720 REM
3730 LET L = 0
3740 LET I = 0
3750 FOR L = 1 TO 1
3760 PRINT "DE ENTRADA A OPCION (1-9)"
3770 FOR I = 1 TO 1
3780 LET A$ = INKEY$
3790 IF A$ = "" THEN I = 0
3800 NEXT I
3810 LET OPCN = VAL(A$)
3820 IF OPCN <1 THEN L = 0
3830 IF OPCN >9 THEN L = 0
3840 NEXT L
3850 RETURN
3860 REM SUBRUTINA $PRIMERA$ (VISUALIZAR MENSAJE)
3870 LET OPCN = 6
3880 PRINT CHR$(12)= REM LIMPIAR PANTALLA
3890 PRINT
3900 PRINT TAB(10); "NO HAY REGISTROS EN"
3910 PRINT TAB(7); "EL ARCHIVO. DEBERA EMPEZAR"
3920 PRINT TAB(8); "POR AGREGAR UN REGISTRO"
3930 PRINT
3940 PRINT TAB(0); "PULSE BARRA ESPACIADORA PARA CONTINUAR"
3950 FOR B = 1 TO 1
3960 IF INKEY$ <> " " THEN B = 0
3970 NEXT B
3980 PRINT CHR$(12): REM LIMPIAR PANTALLA
3990 RETURN
4000 REM SUBRUTINA $EJECUT$
4010 REM
4020 IF OPCN = 6 THEN GOSUB 10000
4030 REM
4040 REM  1 ES $ENCREG$
4050 REM  2 ES $ENCNOM$
4060 REM  3 ES $ENCIUD$
4070 REM  4 ES $ENCINI$
4080 REM  5 ES $LISREG$
4090 IF OPCN = 6 THEN GOSUB 10000
4100 REM  7 ES $MODREG$
4110 REM  8 ES $BORREG$
4120 IF OPCN = 9 THEN GOSUB 11000
4130 REM
4140 RETURN
10000 REM SUBRUTINA $INCREG$
10010 PRINT CHR$(12): REM LIMPIAR PANTALLA
10020 INPUT "DE ENTRADA AL NOMBRE"; NOMCAM$ (TAMA)
10030 INPUT "DE ENTRADA A LA CALLE"; CLLCAM$ (TAMA)
10040 INPUT "DE ENTRADA A LA CIUDAD"; CIUCAM$ (TAMA)
10050 INPUT "DE ENTRADA A LA PROVINCIA"; PROCAM$ (TAMA)
10060 INPUT "DE ENTRADA AL NUMERO DE TELEFONO"; TELCAM$
      (TAMA)
10070 LET RMOD = 1: REM ESTABLECIDA BANDERA 'REGISTRO
      MODIFICADO
10080 LET INDCAM$ (TAMA) = STR$ (TAMA)
10090 LET TEST$ = ""
10100 GOSUB 10200: REM $MODNOM$
10110 LET OPCN = 0
10120 LET TAMA = TAMA +1
10130 REM
10140 REM
10150 RETURN
10200 REM RUTINA $MODNOM$
10210 REM CONVIERTE CONTENIDO DE NOMCAM$ A MAYUSCULAS,
10220 REM QUITA SOBRANTE Y ALMACENA EN EL ORDEN:
10230 REM APELLIDO+ESPACIO+NOMBRE DE PILA EN MODCAM$
10240 REM
10250 LET N$ = NOMCAM$ (TAMA)
10260 FOR L = 1 TO LEN(N$)
10270 LET TEMP$ = MID$(N$,L,1)
10280 LET T = ASC(TEMP$)
10290 IF T >= 97 THEN T = T - 32
10300 LET TEMP$ = CHR$(T)
10310 LET P$ = P$ + TEMP$
10320 NEXT L
10330 LET N$ = P$
10340 REM LOCALIZAR ULTIMO ESPACIO
10350 FOR L = 1 TO LEN(N$)
10360 IF MID$(N$,L,1) = " " THEN S = L
10370 NEXT L
10380 REM QUITAR SOBRANTE Y ALMACENAR NOMBRE DE PILA
10390 REM EN CNOM$
10400 FOR L = 1 TO S - 1
10410 IF ASC(MID$(N$,L,1)) > 64 THEN CNOM$ = CNOM$ +
      MID$(N$,L,1)
10420 NEXT L
10430 REM QUITAR SOBRANTE Y ALMACENAR APELLIDO
10440 REM EN SNOM$
10450 FOR L = S + 1 TO LEN(N$)
10460 IF ASC(MID$(N$,L,1)) > 64 THEN SNOM$ = SNOM$ +
      MID$(N$,L,1)
10470 NEXT L
10480 LET MODCAM$ (TAMA) = SNOM$ + " " + CNOM$
10490 LET P$ = "": LET N$ = "": LET SNOM$ = "": LET
      CNOM$ = ""
10500 LET P$ = "": LET N$ = "": LET SNOM$ = "": LET
      CNOM$ = ""
10510 RETURN
11000 REM SUBRUTINA $SAPROG$
11010 REM CLASIFICA Y GUARDA ARCHIVO
11020 REM SI ALGUN REGISTRO HA SIDO
11030 REM MODIFICADO (RMOD = 1)
11040 REM
11050 IF RMOD = 0 THEN RETURN
11060 REM
11070 GOSUB 11200: REM $CLSREG$
11080 REM
11090 GOSUB 12000: REM $GRDREG$
11100 RETURN
11200 REM SUBRUTINA $CLSREG$
11210 REM CLASIFICA TODOS LOS REGISTROS EN ORDEN
      ALFABETICO
11220 REM MEDIANTE MODCAM$ Y ACTUALIZA INDCAM$
11230 REM
11240 REM
11250 LET S = 0
11260 FOR L = 1 TO TAMA - 2
11270 IF MODCAM$(L) > MODCAM$(L + 1) THEN GOSUB 11350
11280 NEXT L
11290 IF S = 1 THEN 11250
11300 REM
11310 REM
11320 LET RMOD = 0: REM LIMPIA BANDERA 'REGISTRO
      MODIFICADO'
11330 REM
11340 RETURN
11350 REM SUBRUTINA "INTERCALAR"
11360 LET TNOMC$ = NOMCAM$(L)
11370 LET TMODC$ = MODCAM$(L)
11380 LET TCLLC$ = CLLCAM$(L)
11390 LET TCIUC$ = CIUCAM$(L)
11400 LET TPROC$ = PROCAM$(L)
11410 LET TTELC$ = TELCAM$(L)
11420 REM
11430 LET NOMCAM$(L) = NOMCAM$(L + 1)
11440 LET MODCAM$(L) = MODCAM$(L + 1)
11450 LET CLLCAM$(L) = CLLCAM$(L + 1)
11460 LET CIUCAM$(L) = CIUCAM$(L + 1)
11470 LET PROCAM$(L) = PROCAM$(L + 1)
11480 LET TELCAM$(L) = TELCAM$(L + 1)
11490 LET INDCAM$(L) = INDCAM$(L + 1)
11500 REM
11510 LET NOMCAM$(L + 1) = TNOMC$
11520 LET MODCAM$(L + 1) = TMODC$
11530 LET CLLCAM$(L + 1) = TCLLC$
11540 LET CIUCAM$(L + 1) = TCIUC$
11550 LET PROCAM$(L + 1) = TPROC$
11560 LET TELCAM$(L + 1) = TTELC$
11570 LET INDCAM$(L + 1) = STR$(L + 1)
11580 LET S = 1
11590 REM
11600 RETURN
12000 REM SUBRUTINA $GRDREG$
12010 REM
12020 REM
12030 OPEN "O",#1, "DAT.AGCO"
12040 REM
12050 FOR L = 1 TO TAMA - 1
12060 PRINT #1, NOMCAM$(L);","; MODCAM$(L);","; CLLCAM$(L);
      ","; CIUCAM$(L)
12070 PRINT #1, PROCAM$(L);",": TELCAM$(L);",";
      INDCAM$(L);
12080 NEXT L
12090 REM
12100 REM
12110 REM
12120 REM
12130 CLOSE #1
12140 REM
12150 RETURN
12500 REM SUBRUTINA $FINARCH$
12510 IF NOMCAM$(L) = "" THEN LET L = 50
12520 IF NOMCAM$(L) = "" THEN RETURN
12530 LET TAMA = TAMA + 1
12540 REM
12550 REM
12560 RETURN
```

Vannevar Bush

El analizador diferencial
Esta máquina fue diseñada para resolver un tipo importante de funciones matemáticas, que se presentan en numerosas áreas de la ciencia y de la ingeniería, conocidas como ecuaciones diferenciales de segundo grado. El método había sido sugerido por primera vez por lord Kelvin y hacía referencia a la alimentación de la salida de un "integrador" (un dispositivo que calcula con precisión el área de una superficie irregular) conectada a la entrada de otro. Sin embargo, la potencia de salida era demasiado débil y no se podía utilizar como entrada; este método no pudo ser aplicado hasta que fueron inventados los amplificadores.
La máquina que Bush construyó en 1931 era totalmente mecánica y constaba de una compleja estructura de engranajes, ejes y motores eléctricos. La entrada y la salida se producían mediante giro de ejes y el problema de feedback fue solventado mediante un amplificador de par.
En la década de los cuarenta, se construyó un analizador diferencial más avanzado utilizando componentes eléctricos, pero la máquina pesaba más de cien toneladas. La salida se producía mediante cinco registros de tipo digital y tenía una precisión de uno a 10 000. Las condiciones iniciales y los parámetros de control se suministraban en cinta de papel perforado



El analizador diferencial de Bush era una calculadora electromecánica que resolvía ecuaciones diferenciales

Muchas personas sostienen la teoría de que el norteamericano Vannevar Bush fue el padre del ordenador. Su contribución más importante al desarrollo de la ciencia informática tuvo lugar en 1931, cuando creó un analizador diferencial mecánico que constituyó el punto de partida de una serie de investigaciones que finalmente conducirían al desarrollo del ordenador digital.

Bush nació cerca de Boston (Massachusetts) el 11 de marzo de 1890 y, siguiendo los pasos de su padre, estudió la carrera de ingeniería. Tras graduarse en 1913, trabajó durante un corto período para la General Electric, antes de aceptar el cargo de profesor adjunto en su antigua escuela. Luego realizó estudios de posgrado en la Universidad de Harvard y en el Massachusetts Institute of Technology (MIT). Durante la primera guerra mundial colaboró con las fuerzas armadas norteamericanas en la creación y el posterior desarrollo de un detector de submarinos.

Cuando Bush ideó y realizó su primer invento, un dispositivo topográfico, aún era un estudiante. El mecanismo, que estaba suspendido entre dos ruedas de bicicleta, calculaba la altura del suelo sobre el cual se desplazaba y ofrecía una representación en forma gráfica del perfil del terreno. Incorporaba también un dispositivo que hacía las funciones de un integrador, puesto que la determinación de la altura de una posición cualquiera requería el conocimiento de todos los valores previos.

En el MIT, Bush fue profesor de transmisión de energía eléctrica, y empezó las investigaciones sobre uno de los principales problemas concernientes al suministro de electricidad: cómo evitar los cortocircuitos que se producen como resultado de un incremento de la demanda de energía en un momento determinado. Las ecuaciones matemáticas que rigen una situación de este tipo habían sido descubiertas a finales del siglo pasado por el científico escocés James Clerk Maxwell (1831-1879). Pero el problema comportaba tal número de ecuaciones simultáneas que era imposible de resolver manualmente, y por eso Bush empezó sus trabajos sobre el diseño de una máquina que llevara a efecto estos complicados cálculos. Se inspiró también en los trabajos de lord Kelvin (1824-1907), autor de una máquina para resolver las ecuaciones matemáticas relativas a la predicción de las mareas.

A principios de la década de los veinte, Bush construyó su primera máquina, a la que dio el nombre de "producto telegráfico". Esta máquina permitía que un operador humano trazara las trayectorias dibujadas en un gráfico (utilizando un potenciómetro: dispositivo que transforma la medida de una posición en un voltaje). Luego, con estas señales eléctricas, se alimentaba un vatímetro especialmente diseñado: el disco giratorio que figura en todos los contadores de consumo de energía eléctrica, el cual registra la cantidad de vatios consumidos mediante la integración de los valores fluctuantes de intensidad y voltaje para dar el "producto".

El éxito de esta máquina en la resolución de un conjunto de ecuaciones simultáneas sugirió la posibilidad de construir un dispositivo que fuera capaz de resolver ecuaciones diferenciales de segundo grado e incluso más difíciles. Las posteriores investigaciones de Bush en este campo condujeron a concluir el primer analizador diferencial en 1931. La máquina constituyó un éxito completo, construyéndose diversas unidades en Gran Bretaña y el resto en Europa. En Norteamérica, la Escuela Moore de Ingeniería Eléctrica de la Universidad de Pennsylvania —que posteriormente construiría el ordenador ENIAC (véase p. 88)— encargó una de ellas. Cuando el "producto telegráfico" de Bush tenía un margen de error del 2 %, el analizador diferencial proporcionaba resultados con una precisión del 99,95 %. Sin embargo, el coste en la mejora de la fiabilidad de este tipo de dispositivos mecánicos se multiplicaba por diez por cada decimal adicional. Con el desarrollo del ordenador digital, en cambio, el coste de una máquina, para un incremento similar en la fiabilidad, sólo se duplicaba.

Bush llegó a ser decano de la escuela de ingeniería y vicepresidente del Instituto Carnegie en 1939; su habilidad en la administración del fondo para la investigación científica que legara el millonario Carnegie dio como resultado que fuera nombrado, al año siguiente, presidente del Comité de Investigaciones para la Defensa Nacional. En este cargo, Bush fue responsable de las investigaciones del Ejército norteamericano durante la segunda guerra mundial y, en particular, influyó para que se autorizara el proyecto Manhattan, que conduciría a la creación de la bomba atómica. Se jubiló en 1955 y se dedicó a sus hobbies. Murió en 1974.

UNION PERFECTA

Así se comportan los periféricos creados por SINCLAIR para SINCLAIR: de forma perfecta. Y es lógico.

Cada vez que SINCLAIR diseña un microordenador, no lo hace de una manera aislada. Simultáneamente crea todos esos periféricos que van a hacer más potente, preciso y útil el microordenador que tiene entre manos.

Periféricos pensados y diseñados para dar un servicio óptimo, pero con un precio razonable, dentro de la filosofía SINCLAIR:

"Hacer la informática accesible a todos".

Por eso cuando creó el ZX 81 vio la necesidad de dotarlo con una ampliación de memoria de 16K RAM para que no quedara pequeño y de una impresora sencilla y barata pero útil y precisa.

Así es la filosofía SINCLAIR. Así son los periféricos de SINCLAIR para SINCLAIR.





PARA JUGAR A LO GRANDE
(INSTANTANEAMENTE)

Presentamos **el Interface 2 ZX** Pensado y diseñado por SINCLAIR para unirse a la perfección con tu microordenador Spectrum.

Si a la hora de elegir tu microordenador optaste por el mejor, es lógico que elijas ahora el Interface 2 ZX

Ya habrás podido deleitarte con la más amplia variedad de juegos existentes para tu Spectrum (la más extensa del mercado). Ahora con el Interface 2 ZX vas a tener más ventajas para tu Spectrum:



- Podrás conectar Joysticks para sacarle, aún, mayor rendimiento a tus mejores juegos y divertirte con aquellos exclusivamente disponibles en **Cartuchos ZX:** correr, saltar, volar... a lo grande. ¡Menuda diferencia!
- Además, al ser cartuchos con memoria ROM, podrás, con tu SPECTRUM de 16 K, jugar con programas hasta ahora reservados para 48 K, sin ampliar la memoria. ¡Vaya ahorro!
- Al conectar el Interface 2 ZX tienes la certeza de poseer un periférico pensado por SINCLAIR para SINCLAIR. Tu microordenador queda a salvo de circuitos poco fiables. ¡Un alivio!
- Al adquirir el Interface 2 ZX y los Cartuchos ZX en la red de Concesionarios Autorizados, podrás exigir la tarjeta de garantía INVESTRONICA, única válida en territorio nacional. ¡Una tranquilidad!

Interface 2 ZX y Cartuchos ZX

Si aún no los tienes
no sabes lo que te pierdes

Solicita una demostración en cualquier Concesionario Autorizado INVESTRONICA.



CENTRAL COMERCIAL: Tomás Bretón, 60
Tel. 468 03 00 Telex: 23399 IYCO E Madrid.
DELEGACION CATALUÑA: Camp, 80 - Barcelona - 22